CHAPTER 2

Literature Survey: Data Clustering Algorithms and Soft Computing

2.1 In a Nutshell

When a soft computing model is developed for data clustering, it first calls for an understanding of the soft computing paradigm of computing. Soft computing provides a common platform for genetic algorithms, neural networks and fuzzy logic to come together. This association has been tapped to evolve a method for data clustering. After an introduction to genetic algorithms and fuzzy logic, a literature survey on the data clustering algorithms available has been carried out. Following the survey of conventional algorithms, it is seen how soft computing techniques have, of late, been successfully applied to data clustering problems. A comparison of the various techniques of clustering is done. Though the long term aim of the algorithm is general clustering, the algorithm developed in its present form, is suited only for sequential data clustering. As this comes under the purview of grouping, the general problem of grouping ordered data is discussed. The conclusion sums up the discussion, making it easy to study the new algorithm developed in this work in the light of the existing algorithms.
2.2 Soft Computing

2.2.1 Hard Computing vs. Soft Computing

The term “soft computing” automatically brings to mind the opposite concept of “hard computing”. Hard computing is what we have been used to, what has been going on, right from the beginning of computational science. All the classical reasoning and modeling approaches that are based on Boolean logic, analytical models and crisp classifications fall in this category. Hard computing comes down hard on precision leaving no room for approximations. This can be computationally expensive, time consuming and sometimes even impossible for application to complex real-life problems because many such problems are typically ill-defined systems, difficult to model and with large solution spaces. In such cases, soft computing comes to the rescue but with a price of compromising on some of the principles of hard computing. It is tolerant of imprecision, uncertainty, partial truth, and approximation. The guiding principle of soft computing is: exploit the tolerance for imprecision, uncertainty, partial truth, and approximation to achieve tractability, robustness and low solution cost. Fig. 2.1 gives a graphical summary of the domains covered by hard computing and soft computing [Bonissone 1997].

![Fig. 2.1: Hard Computing vs. Soft Computing](image)

2.2.2 Soft Computing Paradigms

Soft Computing is a recently coined term describing the symbiotic use of many emerging computing disciplines. The principle components of soft computing are the computing disciplines of fuzzy logic, probabilistic reasoning, neural networks and genetic algorithms. Soft computing brings these disciplines under a common umbrella. It is a
partnership in which each of the partners contributes a distinct methodology for addressing problems in its domain. In this perspective, the principal constituent methodologies in soft computing are complementary rather than competitive.

Soft Computing combines the versatility of Fuzzy Logic to represent qualitative knowledge, with the data-driven efficiency of Neural Networks to provide fine-tuned adjustments via local search, with the ability of Genetic Algorithms to perform efficient coarse-granule global search. Leveraging the best features of these algorithms results in the development of hybrid algorithms that are superior to each of their underlying components.

Fuzzy Logic, first presented by Zadeh [1965], introduced the idea of partial set membership, in which we can translate our qualitative knowledge about the problem to be solved. Fuzzy logic’s main characteristic is the robustness of its interpolative reasoning mechanism. Fuzzy logic enables us to translate and embed empirical, qualitative knowledge about the problem to be solved into reasoning systems capable of performing approximate pattern matching and interpolation. Fuzzy logic however does not have adaptation or learning features, since it lacks the mechanism to extract knowledge from existing data. Fuzzy logic systems are not able to learn from examples of input-output pairs, in a typical supervised mode.

Probabilistic Reasoning (PR) such as Bayesian Belief Networks, based on the original work of Bayes [1763] and Dempster-Shafer’s theory of belief, independently developed by Dempster [1967] and Shafer [1976], gives us the mechanism to evaluate the outcome of systems affected by randomness or other types of probabilistic uncertainty. PR’s main characteristic is its ability to update previous outcome estimates by conditioning them with newly available evidence.

Artificial Neural Networks (ANN), first explored by Rosenblatt [1958], Widrow and Hoff [1960], are computational structures that can be trained to learn patterns from examples. By using a training set that samples the relation between inputs and outputs, and a back-propagation type of algorithm introduced by Werbos [1974], ANNs give us a supervised learning algorithm that performs fine-granule local optimization.
Genetic Algorithms (GA), proposed by Holland [1975], give us a way to perform randomised global search in a solution space. In this space, a population of candidate solutions, encoded as chromosomes, is evaluated by a fitness function in terms of its performance. The best candidates evolve and pass some of their characteristics to their offspring.

As the theme of this thesis is a clustering algorithm based on genetic algorithms and the fuzzy systems concept, these topics have been covered in more detail.

2.3 Genetic Algorithms

Traditional calculus-based methods are local in scope and depend on the existence of derivatives. Hence, they are unsuitable for discontinuous, multi-modal and noisy search spaces. Other conventional methods like enumerative schemes and random search are also not robust. Genetic algorithms are theoretically and empirically proven to provide robust search in complex spaces.

Genetic algorithms are search algorithms based on the mechanics of natural genetics. They follow an iterative procedure that consists of a constant-size population of individuals, each one represented by a finite string of symbols, encoding a possible solution in a given problem space. This space, referred to, as the search space, comprises all possible solutions to the problem at hand. Generally speaking, the genetic algorithm is applied to spaces which are too large to be exhaustively searched. The symbol alphabet used is often binary, though other representations have also been used, including character-based encodings, real-valued encodings and tree representations.

While solving a problem, we are usually looking for some solution which will be the best among others. The space of all feasible solutions (the set of solutions among which the desired solution resides) is called search space (also state space). Each point in the search space represents one possible solution. Each possible solution can be "marked" by its
value (or fitness) for the problem. With GA we look for the best solution among a number of possible solutions - represented by one point in the search space.

2.3.1 Basic Genetic Algorithm

An outline of the basic genetic algorithm is given below.

1. [Start] Generate random population of \( n \) chromosomes (suitable solutions for the problem)

2. [Fitness] Evaluate the fitness \( f(x) \) of each chromosome \( x \) in the population

3. [New population] Create a new population by repeating following steps until the new population is complete
   
   a) [Selection] Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)
   
   b) [Crossover] With a crossover probability cross over the parents to form new offspring (children). If no crossover was performed, offspring is the exact copy of parents.
   
   c) [Mutation] With a mutation probability mutate new offspring at each locus (position in chromosome).
   
   d) [Accepting] Place new offspring in the new population

4. [Replace] Use new generated population for a further run of the algorithm

5. [Test] If the end condition is satisfied, stop, and return the best solution in current population

6. [Loop] Go to step 2

Genetic Algorithms search from a population of points, not a single point. They work with a coding of the parameter set, not the parameters themselves. Objective functions decide the course of the search instead of derivatives or other auxiliary knowledge. Probabilistic transition rules are used instead of deterministic rules. These are the features that contribute to the robustness of genetic algorithms.

2.3.2 Multi-Objective Genetic Algorithm

Many problems in nature have several objectives (sometimes conflicting) to be satisfied. These are handled by either converting the multiple objectives to one by using weighting functions or by using one objective to optimize and transforming the others into
constraints [Michalewicz, Z. 1999]. But in these methods there can be loss of significant trade-off information and the inability to search for a global optimum [Andersson and Krus 2001].

Another approach is to consider the solutions that represent the optimal trade-offs or the set of locally best alternatives. The objective is to find a set of solutions that are globally non-dominated i.e. the solution is such that there are no feasible solutions that would better one criterion without simultaneously worsening at least one other criterion. This set of non-dominated solutions is called the Pareto-optimal set or the P-optimal set named after Vifredo Pareto who generalized this concept of optimality [Fonseca and Fleming 1995].

Multi-objective optimization has to work with a set of solutions. Since GAs also manipulate a set of solutions, they are naturally suited for this kind of problem [Fonseca and Fleming 1993;Horn et al. 1994]. Multi-Objective Genetic Algorithm (MOGA) proposed by Fonseca and Fleming [1993] uses a ranking scheme where the rank of a certain individual corresponds to the number of individuals in the current population by which it is dominated. In the Non-dominated Sorted GA (NSGA) by Srinivas and Deb [1994], a layer-wise pareto ranking is done. The non-dominated individuals are sorted and removed. Evolutionary multi-objective genetic algorithms attempt to devise clever ways to produce non-dominated solutions.

2.4 Fuzzy Logic

2.4.1 Background

Fuzzy logic, the logic of approximation introduced by Zadeh was developed by Mamdani and Assilian in 1974 to regulate a steam engine [Dutta, 1993]. In 1985, Bell Laboratories developed the first fuzzy logic chip. Fuzzy logic can be implemented in hardware, software or a combination of both. Most of the fuzzy logic applications are in the area of process control (air conditioning, cruise control, ship boiler parameters, washing machines etc.). Today, fuzzy logic is a success story of applied artificial intelligence technology.
Fuzzy logic incorporates a simple, rule-based approach to solving a control problem rather than attempting to model a system mathematically. The model is empirically-based, relying on operator experience rather than their technical understanding of the system. The use of descriptive language to deal with input data more like a human operator makes it robust and simple to develop and often works when first implemented with little or no tuning.

2.4.2 Design Process of a Fuzzy Logic System

The generic fuzzy reasoning system has a set of design steps. By applying these steps to the problem domain, a working fuzzy system can be designed. A fuzzy logic system is an input-output system which when presented with one or more inputs yields one or more output parameters. The main design steps are :-

1) Data Processing
The range and membership functions for the input and output variables are defined. The rule base consisting of if-then rules associating inputs with outputs is designed.
The membership function is a graphical representation of the magnitude of participation of each input. It associates a weighting with each of the inputs that are processed, defines functional overlap between inputs, and ultimately determines an output response. They are generally triangular, bell-shaped or trapezoidal in shape.

2) Fuzzification of Inputs
When an input is given to the system in the form of a crisp value for the input variable, the degree to which the crisp input belongs to all the fuzzy sets corresponding to that variable is calculated.

3) Fuzzy Inference Mechanism
The fuzzified inputs are applied to the antecedents of the rule base. Different rules get activated based on the activation values of the input variables. The consequent parts of some rules are satisfied while others are not. The conclusions are combined to form a logical sum for each membership function. The inference process determines the output firing strengths (0 to 1) of each output membership function.
4) Defuzzification

Defuzzification, the reverse of fuzzification, converts the fuzzy set confidences into real numbers. The consequents of the fired rules are combined by a defuzzification method to give crisp output values to external world. Commonly used methods for defuzzification are the maximum method, centroid method and the weighted average method. The maximum method gives the output which has the highest membership function. The most commonly used centroid method [Sugeno 1985] combines the membership functions and calculates the centroid of the area. The third method finds the weighted average of the outputs from the rules in the rulebase.

In the next sub-section, fuzzy variable design is dealt with, in detail, as the algorithm developed in this work is also applied to design of membership function generation.

2.4.3 Fuzzy Variable Design

Fuzzy variables are the building blocks of fuzzy systems. A fuzzy variable is characterized by its membership function. The two main issues in the design of membership functions are that of finding the appropriate number of membership functions (number of mapping categories) and determining the shape of the membership functions. Another issue is the determination of the universe of discourse which is the extent of validity of the fuzzy inference rules on each variable domain. Outside the domain, the performance of the fuzzy system will be unpredictable or ill defined.

The two main approaches to membership function design are data-driven design and linguistic design [Berkan and Sheldon 1997]. The data-driven approach utilizes a set of numerical data that represents the inference mechanism. For example, with a data set from a power plant with the plant variables and the operator’s control actions, a fuzzy inference engine automating the operator’s decisions in the form of IF-THEN rules can be developed. Data-driven design assumes that the decision boundaries are known or they are easily extractable from the data set. That is, it is known which point in the input product space belongs to which output class. When such data is not available, the design becomes a linguistic design. Linguistic design uses heuristic interpretations due to the lack of appropriate data or because of the nature of the problem. In both approaches, the concept of decision boundaries plays a major role. Decision boundaries indicate which
regions in the input space correspond to which regions in the output space in an input-output mapping process.

2.4.3.1 Data Driven Approach

The two forms of membership function development use the point membership function and the probabilistic membership functions.

To understand point-membership functions, consider a product space between two input variables. Let the relation among the inputs $I_1$, $I_2$ and $O$ be defined by a single point. Fuzziness around the point may be defined by a pyramid (or a cone) which produces triangular projections on both product spaces. The projected triangles are the membership functions.

When there are $N$ points describing the relation of interest, point membership function development is repeated for each point on the product space. In general, $N$ crisp points on the input-output product space require $N$ mapping rules with $N$ pairs of point membership functions for two antecedent variables.

![Fig. 2.2: Points on the product space surrounded by pyramid surface extensions representing fuzzy regions](image)

When all output classes are distinct, the exhaustive solution can be taken. Otherwise the exhaustive solution should be reduced to an approximated solution using fewer rules by combining some of the point-membership functions.
In probabilistic membership function design, probability estimates are assigned. The designer must identify whether the problem requires truth characterization or truth categorization. Truth characterization applies to problems in which the outcome of an event is estimated by repeated measurements. Truth categorization applies to problems in which the category of an event is estimated by judgment (human expertise and commonsense reasoning). Most of the fuzzy systems employ truth categorization. A membership function can be defined by the frequency of occurrence data that are obtained by repeated measurements of the same event. An example is shown in Fig. 2.3 for three statistically independent events A, B and C on the same universe of discourse given by \( x \in [0, 21] \). Each event is tracked independently until a probabilistic distribution is developed.

![Designing a probabilistic fuzzy variable](image)

Fig 2.3 Designing a probabilistic fuzzy variable

Another method is that of minimum entropy. There are probabilistic methods in literature to partition the product space provided the number of categories (decision regions) is known. One simple method is based on finding a cluster with minimum entropy and using the cluster boundaries as the intersecting points of the membership functions.

### 2.4.3.2 Linguistic Approach

In linguistic fuzzy variable design, the articulation of knowledge in terms of fuzzy rules is the initial requirement by which the number of mapping categories is determined.
Thereafter, only the location and shapes of the membership functions are to be determined.

To determine the location of membership functions, the key element is the decision boundaries embedded in the articulation of rules. Consider the fuzzy rules:

**IF Load IS Too-heavy THEN Bridge WILL Collapse**
**IF Load IS Moderate THEN Bridge WILL Sustain**

The decision boundary between Too-heavy and Moderate on the universe of Load cannot be guessed during design and therefore linguistic design is not suitable in this case. One can use linguistic design when the stringency on the accuracy of the decision boundary is less.

For determining shapes, semantics are the driving force. There can be semantically constrained and unconstrained predicates. Predicates are labels that categorize different regions of the universe of discourse. A semantically unconstrained predicate implies infinite degrees of freedom in the design of membership functions. Measures of strength, quantity and quality fall into this category (high, small, medium, heavy). On the other hand, semantically constrained predicates are constrained by language like measures of comparison (less than, almost, quite, very) which are also known as linguistic hedges. Hedges modify predicates whereas predicates modify fuzzy variables. Membership function design depends on individual style, experience etc. Validation of such designs is difficult and generally an iterative approach to reduce cost is followed.

### 2.5 Data Clustering

A cluster is made up of a number of similar objects grouped together. Everitt [1974] defines a cluster as a set of entities which are alike and entities from different clusters are not alike. The clustering process has to find the hidden structure in the data. The typical components of a clustering task [Jain and Dubes 1988] are :-

i) pattern representation

ii) proximity/similarity measures

iii) clustering algorithms
iv) cluster evaluation/validation

v) interpretation of results

Most clustering models assume that all data items are represented by n-dimensional feature vectors. This first step therefore involves choosing appropriate features, preprocessing and feature extraction on data items to measure the values of the chosen feature set. It will often be desirable to choose a subset of all the features available, to reduce the dimensionality of the problem space. This step often requires a good deal of domain knowledge and data analysis.

The second step is to find pattern proximity which is usually measured by a distance function defined on pairs of patterns. A variety of distance measures are in use [Anderberg 1973; Jain and Dubes 1988; Diday and Simon 1976]. 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 [Michalski and Stepp 1983].

The main event is the clustering algorithm (step 3) which will be covered in detail in the next section. All clustering algorithms will, when presented with data, produce clusters regardless of whether the data contain clusters or not. If the data does contain clusters, some clustering algorithms may obtain ‘better’ clusters than others. Cluster validity analysis is the assessment of a clustering procedure’s output and is performed after clustering.

The whole idea of grouping the data is to make valuable inferences from the clusters which were otherwise not possible on the original data. This is the last step, the results of which form the final output of the clustering task. Such inferences will be problem specific.

Clustering algorithms being the main theme of this thesis, a survey of the various algorithms available in literature will be done [Hartigan 1975]. The other tasks of clustering are problem dependent and hence need to be considered at the application stage of the algorithm only. The discussion on clustering algorithms is adapted from [Jain et al. 1999; Jain and Dubes 1988] and other sources. Clustering is a special kind of
classification. The classification tree by Lance and Williams [1967] is shown in Fig. 2.4. The nodes of the tree are described below.

a) Exclusive versus non-exclusive – An exclusive classification is a partition of the set of objects such that each object belongs to exactly one cluster. Nonexclusive or overlapping classification can assign an object to several classes.

b) Intrinsic versus Extrinsic – An intrinsic classification uses only the similarity matrix (consisting of similarity measures between the data items) and is called ‘unsupervised learning’ because no category labels denoting an apriori partition of the objects are used. Extrinsic classification uses both category labels as well as the information from the similarity matrix.

c) Hierarchical versus partitional – These belong to the exclusive and intrinsic type of classification problems. They differ in the type of structure imposed on the data. A hierarchical classification is a nested sequence of partitions whereas a partitional classification is a single partition of the data set.

![Classification Tree](image)

**Fig. 2.4: Classification tree of clustering algorithms**

The taxonomy of data clustering algorithms divides clustering methods into two broad categories – hierarchical or partitional. There are also other algorithms that do not fall into either of these two categories. They are studied separately.
2.5.1 Hierarchical Approach

Hierarchical clustering builds a hierarchy of clusters. The traditional representation of this hierarchy is a tree, with individual elements at one end and a single cluster with every element at the other (Fig. 2.5). A hierarchical algorithm yields a dendrogram representing the nested grouping of patterns and similarity levels at which groupings change.

Agglomerative algorithms begin at the top of the tree, whereas divisive algorithms begin at the bottom. (In the figure, the arrows indicate an agglomerative clustering).

![Hierarchical Clustering Diagram](image)

Fig. 2.5: Hierarchical Clustering

Cutting the tree at a given height will give a clustering at a selected precision. In the above example, cutting after the second row will yield clusters \{a\} \{b c\} \{d e\} \{f\}. Cutting after the third row will yield clusters \{a\} \{b c\} \{d e f\}, which is a coarser clustering with fewer clusters.

The agglomerative hierarchical clustering method builds the hierarchy from the individual elements by progressively merging clusters. There are six elements \{a\} \{b\} \{c\} \{d\} \{e\} and \{f\}. The first step is to determine which elements to merge in a cluster. Generally, the two closest elements should be merged, therefore a distance function $d$ is defined between elements.
Suppose the two closest elements are \( b \) and \( c \) and they are merged, then the resultant clusters are \( \{a\}, \{b, c\}, \{d\}, \{e\} \) and \( \{f\} \). To merge them further, the distance between clusters \( \{a\} \) and \( \{b, c\} \) has to be defined. The distance between two clusters can be defined in different ways. It can be the maximum distance between two elements drawn one each from each cluster (called complete linkage clustering). Similarly, the minimum distance version is called single linkage clustering.

Each agglomeration occurs at a greater distance between clusters than the previous agglomeration, and one can decide to stop clustering either when the clusters are too far apart to be merged or when there is a sufficiently small number of clusters.

In complete-link (or complete linkage) hierarchical clustering, at each step the two clusters whose merger has the smallest diameter (or: the two clusters with the smallest maximum pair-wise distance) are merged. In single-link (or single linkage) hierarchical clustering, the two clusters whose two closest members have the smallest distance (or: the two clusters with the smallest minimum pair wise distance) are merged.

The clusters obtained by the complete link algorithm are more compact than those obtained by the single-link algorithm. The single-link algorithm is more versatile than the complete-link algorithm, otherwise. For example, the single-link algorithm can extract the concentric clusters but the complete-link algorithm cannot.

### 2.5.2 Partitional Approach

A partitional clustering algorithm obtains a single partition of the data instead 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.

The \( k \)-means is the simplest and most commonly used partitional algorithm employing a squared error criterion [McQueen 1967]. It starts with a random initial partition and keeps reassigning the patterns to clusters based on the similarity between the pattern and the cluster centers until a convergence criterion is met (e.g., there is no reassignment of
any pattern from one cluster to another, or the squared error ceases to decrease significantly after some number of iterations). The $k$-means algorithm is popular because it is easy to implement, and its time complexity is linear in $n$ where $n$ is the number of patterns. A major problem with this algorithm is that it is sensitive to the selection of the initial partition and may converge to a local minimum of the criterion function value if the initial partition is not properly chosen.

The $k$-means algorithm assigns each point to the cluster whose centroid is nearest. The centroid is the point generated by computing the arithmetic mean for each dimension separately for all the points in the cluster. The basic structure of the algorithm is given.

1) Randomly generate $k$ clusters and determine the cluster centers or directly generate $k$ seed points as cluster centers.

2) Assign each point to the nearest cluster center.

3) Recompute the new cluster centers.

4) Repeat until the convergence criterion is met

The main advantages of this algorithm are its simplicity and speed, which allows it to run on large datasets. Yet it does not systematically yield the same result with each run of the algorithm. Rather, the resulting clusters depend on the initial assignments. The $k$-means algorithm maximizes inter-cluster (or minimizes intra-cluster) variance, but does not ensure that the solution given is not a local minimum of variance.

Hierarchical algorithms are more versatile than partitional algorithms. For example, the single-link clustering algorithm works well on data sets containing non-isotropic clusters including well-separated, chain-like, and concentric clusters, whereas in a typical partitional algorithm such as the $k$-means algorithm works well only on data sets having isotropic clusters [Nagy 1968]. On the other hand, the time and space complexities [Day 1992] of the partitional algorithms are typically lower than those of the hierarchical algorithms. It is possible to develop hybrid algorithms [Murty and Krishna 1980] that exploit the good features of both categories.
2.5.3 Graph-Theoretic Approach

The best-known graph-theoretic divisive clustering algorithm is based on construction of the *minimal spanning tree* (MST) of the data [Zahn 1971] and then deleting the MST edges with the largest lengths to generate clusters. Fig. 2.6 depicts the MST obtained from nine two-dimensional points. By breaking the link labeled CD with a length of 6 units (the edge with the maximum Euclidean length), two clusters ({A, B, C} and {D, E, F, G, H, I}) are obtained. The second cluster can be further divided into two clusters by breaking the edge EF, which has a length of 4.5 units.

![Minimum Spanning Tree](image)

Fig. 2.6: Minimum Spanning Tree

The hierarchical approaches are also related to graph-theoretic clustering. Single-link clusters are subgraphs of the minimum spanning tree of the data [Gower and Ross 1969]. Complete-link clusters are maximal complete subgraphs.

2.5.4 Grid-based Algorithms

A number of clustering algorithms have been presented for spatial data, known as grid-based algorithms. These algorithms quantise the space into a finite number of cells and then do all operations on the quantised space. STING (Statistical Information Grid-based method) is representative of this category. It divides the spatial area into rectangular cells using a hierarchical structure. STING [Wang et al 1997] goes through the data set and computes the statistical parameters (such as mean,
variance, minimum, maximum and type of distribution) of each numerical feature of the objects within cells. Then it generates a hierarchical structure of the grid cells so as to represent the clustering information at different levels.

**WaveCluster** [Sheikholeslami et al 1998] is a grid-based algorithm based on signal processing techniques (wavelet transformation). A multidimensional grid structure is imposed onto the data space. Each grid cell summarizes the information of a group of points that map into the cell. Then it uses a wavelet transformation to transform the original feature space. In wavelet transforms, convolution with an appropriate function results in a transformed space where the natural clusters in the data become distinguishable. Thus, the clusters can be identified by finding the dense regions in the transformed domain.

### 2.6 Clustering Algorithms: Soft Computing Approach

The soft computing paradigm of computing has contributed to the theory of clustering by the application of approximate reasoning to clustering.

#### 2.6.1 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 output of such algorithms is a clustering, but not a partition.

Fuzzy set theory was initially applied to clustering by Ruspini [1969]. In fuzzy clustering [Bezdek 1981], each point does not pertain to a given cluster, but has a degree of belonging to a certain cluster, as in fuzzy logic. For each point \( x \) we have a coefficient giving the degree of being in the \( k \)th cluster \( u_k(x) \). Usually, the sum of those coefficients has to be one, so that \( u_k(x) \) denotes a probability of belonging to a certain cluster:
The most popular fuzzy clustering algorithm is the fuzzy $c$-means (FCM) algorithm. With fuzzy $c$-means, the centroid of a cluster is computed as being the mean of all points, weighted by their degree of belonging to the cluster, that is

$$centre_k = \frac{\sum_x u_k(x)x}{\sum_x u_k(x)}$$

The degree of being in a certain cluster is the inverse of the distance to the cluster,

$$u_k(x) = \frac{1}{d(centre_k, x)}$$

Then the coefficients are normalized and fuzzified with a real parameter $m > 1$ so that their sum is one.

$$u_k(x) = \frac{1}{\sum \left(\frac{d(centre_k, x)}{d(centre_j, x)}\right)^{m-1}}$$

The fuzzy $c$-means algorithm is greatly similar to the $k$-means algorithm; when $m \rightarrow 1$, the algorithms are equivalent: Even though it is better than the hard $k$-means algorithm at avoiding local minima, FCM can still converge to local minima of the squared error criterion.

### 2.6.2 Artificial Neural Networks in Clustering

Competitive (or winner–take–all) neural networks [Jain and Mao 1996], Kohonen’s learning vector quantization (LVQ), self-organizing map (SOM) and adaptive resonance theory models (ART) have been used to cluster input data. The SOM gives an intuitively appealing two-dimensional map of the multidimensional data set, and it has been successfully used for vector quantization and speech recognition [Kohonen 1989]. However, the SOM generates a suboptimal partition if the initial weights are not chosen properly. 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. The system is said to be stable if no pattern in the training data changes its category after a finite number of learning iterations. This problem is closely associated with the problem of plasticity, which is the ability of the algorithm to adapt to new data. For stability, the learning rate should be decreased to zero as iterations progress and this affects the plasticity.

The ART models are supposed to be stable and plastic [Carpenter and Grossberg 1990]. However, ART nets are order-dependent; that is, different partitions are obtained for different orders in which the data is presented to the net. Also, the size and number of clusters generated by an ART net depend on the value chosen for the vigilance threshold, which is used to decide whether a pattern is to be assigned to one of the existing clusters or start a new cluster. Further, both SOM and ART are suitable for detecting only hyperspherical clusters [Hertz et al. 1991]. All these ANNs use a fixed number of output nodes which limit the number of clusters that can be produced.

2.6.3 Evolutionary Approaches for Clustering

Candidate solutions to the clustering problem are encoded as chromosomes. A solution is a valid k-partition of the data. The most commonly used evolutionary operators are: selection, recombination, and mutation. Each transforms one or more input chromosomes into one or more output chromosomes. A fitness function evaluated on a chromosome determines a chromosome’s likelihood of surviving into the next generation.

The best-known evolutionary techniques are genetic algorithms (GA) [Holland 1975; Goldberg 1989], evolution strategies (ES) [Schwefel 1981] and evolutionary programming (EP) [Fogel et al. 1965]. Out of these three approaches, GAs have been most frequently used in clustering. GAs represent points in the search space as binary strings, and rely on the crossover operator to explore the search space. Mutation is used in GAs for the sake of completeness, that is, to make sure that no part of the search space is left unexplored. ESs and EP differ from the GAs in solution representation and type of the mutation operator used; EP does not use a recombination operator, but only selection and mutation. Each of these three approaches has been used to solve the clustering problem by viewing it as a minimization of the squared error criterion.
One of the earliest paper on the use GAs for clustering is by Raghavan and Birchand [1979], where a GA was used to minimize the squared error of a clustering. Here, each point or chromosome represents a partition of $N$ objects into $K$ clusters and is represented by a K-ary string of length $N$. For example, consider six patterns A, B, C, D, E, and F. The six-bit binary (K=2) string, 101001 corresponds to placing the six patterns into two clusters. It represents a two-partition where the two clusters are \{AC,F\} and \{B,D,E\}. There are $K!$ different chromosomes corresponding to each $K$-partition of the data. Hence, for $K$ clusters, the effective search space is increased by a factor of $K!$.

A GA-based clustering algorithm [Maulik and Bandhopadhyay, 2000] attempts to get over the drawbacks of the k-means algorithm converging to sub-optimal values by real-coded GA. Each string is a sequence of real numbers representing the $K$ cluster centres. For an $N$-dimensional space, the length of a chromosome is $N*K$ words, where the first $N$ positions (or, genes) represent the $N$ dimensions of the first cluster centre, the next $N$ positions represent those of the second cluster centre, and so on. The algorithm is a GA implementation of the k-means algorithm.

However, all evolutionary approaches suffer from sensitivity to control parameter selection. For each specific problem, one has to tune the parameter values to suit the application. Evolutionary approaches are recommended only when the data set is small, due to large execution times. Only the k-means algorithm and its ANN equivalent, the Kohonen net have been applied on large data sets. Use of domain knowledge to guide the clustering process could be one way to get around the problem.

Hybrid genetic algorithms incorporating problem-specific heuristics were reported to give good clustering results [Jones and Beltramo 1991; Davis 1991]. The evolutionary approach belongs to the class of stochastic search techniques that generate a near-optimal solution and guarantee asymptotic convergence to optimality. Other stochastic search techniques used for clustering are Simulated Annealing [Klein and Dubes 1989] and Tabu search [Al-Sultan 1995]. Extensive studies dealing with comparative analysis of different clustering methods [Dubes and Jain 1976] suggest that there is no general strategy which works equally well in different problem domains.
2.7 Sequential Data Clustering

Data collected from the external world can be of two types. When the order of the data is important, it is called sequentially correlated data. Otherwise, the data is called cross section data. Most of the dynamic data (like temperature profile of a furnace) are sequentially correlated with time as the independent variable. The algorithm developed in this work has been applied for sequential data clustering. By sequential data, we mean data ordered according to some criterion. This special case of clustering comes under the class of problems covered by grouping theory. Constraints can be imposed to allow only contiguous partitions over some variable or on data-sets that are ordered a priori. For example, time series segmentation consists of finding homogeneous segments that are contiguous in time.

Grouping is defined as the partitioning, or explicit segmentation, of a set of data into homogeneous groups that can be explained by a stochastic model. Explicit segmentation is the partitioning of data into homogeneous regions by specifying cut-points. An ordered set of K numbers \( \{a_i : i = 0..K - 1\} \) can be partitioned into G contiguous groups in \( \binom{K-1}{G-1} \) ways. We only consider contiguous partitions since we assume that the data has been ordered a priori. An expression for the total number of partitions can be calculated.

- Minimum number of partitions = 1, \( G = 1 \)
- Maximum number of partitions = K, \( G = K \)

The number of partitions can vary between 1 and K i.e. \( 1 \leq G \leq K \)

Total number of partitions = \( \binom{K-1}{0} + \binom{K-1}{1} + \ldots + \binom{K-1}{K-1} \)

\[ = 2^{K-1} \quad \text{(sum of binomial coefficients)} \]

For a set of 10 numbers, the number of groups will be 512 but this number becomes \( 5 \times 10^{29} \) for a data set of 100 numbers. Due to the exponential increase in the size of the search space, exhaustive search can be used only for very small data sets.

2.7.1 Fisher’s Solution

For the ordered set of K data items and for a given G, Fisher's solution to the grouping problem was to search for the contiguous partition determined by G -1 cut-points that minimised the distance, D:

\[ D = \sum_{i=0}^{K-1} (a_i - \bar{a}_i)^2 \]

where \( a_i \) represents the arithmetic mean of the a's assigned to the group in which \( a_i \) is assigned. For a given G, the partition which minimises D is called an optimal or least squares partition. Though Fisher was concerned with grouping normally distributed data, his techniques, can be applied to other models.

The exhaustive search algorithm used to find the optimal partition is based on the “Sub-optimisation Lemma”:

Lemma: If \( A_1:A_2 \) denotes a partition of set A into two disjoint subsets \( A_1 \) and \( A_2 \), if \( P_1^* \) denotes a least squares partition of \( A_1 \) into \( G_1 \) subsets and if \( P_2^* \) denotes a least squares partition of \( A_2 \) into \( G_2 \) subsets; then, of the class of sub-partitions of \( A_1:A_2 \) employing \( G_1 \) subsets over \( A_1 \) and \( G_2 \) subsets over \( A_2 \) a least squares sub-partition is \( P_1^* : P_2^* \).

This lemma is possible due to the additive nature of the distance measure. The algorithm based on this lemma is an example of a Dynamic Programming Algorithm (DPA) and is computable in polynomial time. The DPA is a general class of algorithms that is used in optimisation problems where the solution is the sum of sub-solutions. The time complexity of Fisher's DPA is:

\[ \forall_{k=1..G_{\text{max}}-1} \forall_{i=k..K-1} \min_{j=k}^{j=k-1} D(k-1, j-1) + D(j, i) = O(G_{\text{max}} . K^2) \]

In practice, \( G_{\text{max}} \ll K \), where \( G_{\text{max}} \) is an upper bound to G.

2.7.2 The Minimum Message Length Method

For a given data set, to find the number of segments G, there is a range of least square partitions to select from. The problem with Fisher’s maximum likelihood solution, based on minimization of squared error is that the value of G must be known apriori, otherwise there is no stopping criterion and the optimal grouping would consist of one data point per group.
Oliver, Baxter and colleagues [Baxter and Oliver 1996; Oliver and Forbes 1997; Oliver, Baxter and Wallace 1998] have applied the information theoretic Minimum Message Length (MML) principle to explicit segmentation. They have implemented and tested an MML based solution to the segmentation of time series data. The MML principle is based on compact coding theory. It provides a criterion for comparing competing hypotheses by encoding both the hypothesis (H) and the data (D) in a two-part message. From coding theory, an event with probability P can be transmitted using an optimal code in a message of -$\log_2(P)$ bits in length. Using Baye’s theorem, it can be shown that the length of the two-part message (MessLength) can be calculated as:

$$\text{MessLength}(H&D) = -\log_2(Pr(H)) - \log_2(Pr(D|H)) \text{ bits.}$$

The receiver of such a message must be able to decode the data. The model with the shortest message length is considered to give the best solution. Oliver, Baxter and others have studied the issue of specifying the cut-point imprecisely. When the boundary between two segments is not well-defined, it is cheaper to use less precision for the cut-point specification.

Fitzgibbon et al. [2000] have used the MML criterion and Fisher’s DPA to perform numerical Bayesian integration (using message lengths) over the cut-point location parameters to give an estimate of the number of segments. This is then used to estimate the cut-point positions and segment parameters by minimizing the MML criterion.

### 2.7.3 Graph-Based Grouping Method

Graphs as seen in Section 2.5, have been used for clustering and grouping [Wu and Leahy 1993; Shapiro and Haralick 1979; Matula 1977] but Amir and Lindenbaum [1998] proposed a generic grouping algorithm based on graph clustering. The method has two components: the grouping cues and the grouping mechanism. Psychologists noticed that humans use some properties to recognize the existence of perceptual structures in a scene and these properties are called grouping cues. The cue evaluation is done for feature pairs. In principle, all feature pairs correspond to a complete underlying graph. An arc connects a pair of nodes (data features). The graph representation of the perceptual grouping evidence is used in the second stage to find best partitions of the graph into groups. Statistical methods, mainly the maximum likelihood criterion is used for the implementation.
2.8 Cluster Validity Indices

The evaluation of a clustered output has several facets. One is actually an assessment of the data domain rather than the clustering algorithm itself - data which do not contain clusters should not be processed at all by a clustering algorithm. For this, the ‘cluster tendency’ has to be studied, wherein the input data are examined to see if there is any merit to a cluster analysis prior to actual clustering. Often this analysis uses a specific criterion of optimality; however, these criteria are usually arrived at subjectively.

Validity assessments are objective [Dubes 1993] and are performed to determine whether the output is meaningful. A clustering structure is valid if it cannot reasonably have occurred by chance. When statistical approaches to clustering are used, validation is accomplished by carefully applying statistical methods and testing hypotheses. Indices used for these methods are discussed in detail in Jain and Dubes [1988] and Dubes [1993].

Maulik and Bandyopadhyay [2002] have evaluated the performance of clustering algorithms with respect to commonly used clustering indices. For a data set with n elements, \( X = \{x_1, x_2, \ldots, x_n\} \), partitioned into k clusters \( (C_1, C_2, \ldots, C_k) \), the Davies-Bouldin (DB) Index [Davies and Bouldin 1979] is a function of the ratio of the sum of within-cluster scatter to between-cluster separation. The scatter within the \( i^{th} \) cluster, \( S_i \) is given by

\[
S_i = \frac{1}{|C_i|} \sum_{x \in C_i} \|x - z_i\|
\]

The distance between cluster \( C_i \) and \( C_j \) is given by

\[
d_{ij} = \|z_i - z_j\| \quad \text{where} \quad z_i \text{ represents the } i^{th} \text{ cluster center.}
\]

DB = \( \frac{1}{K} \sum_{i=1}^{K} R_i \) where \( R_i = \max_{j \neq i} \left\{ \frac{S_i + S_j}{d_{ij}} \right\} \)

The objective is to minimize the DB index for achieving proper clustering.

Dunn’s index [Dunn 1973] calculates a cluster validity measure by defining the diameter of a cluster and the distance between clusters. The diameter of a cluster is defined as the
maximum of all the distances between all pairs of points in the cluster. The distance between two clusters is the minimum distance between the clusters obtained by taking distances between all pairs of points such that if one point belongs to one cluster, the other point belongs to the other cluster. The minimum of the distance to diameter ratio taken over all the K clusters gives the Dunn’s index. Larger values of Dunn’s index correspond to good clusters, and the number of clusters that maximizes the index is taken as the optimal number of clusters.

The CH index [Calinski and Harabasz 1974] is computed using the between cluster and within cluster scatter matrices. The index for n data points and K clusters is computed as

\[
\frac{\text{trace}B}{(K-1)} \times \frac{1}{\text{trace}W/(n-K)}
\]

where B and W are within cluster scatter matrices.

The DB index, Dunn’s index and the CH index are three commonly used indices to measure the adequacy of a structure recovered through cluster analysis. Since clustering algorithms define clusters that are not known a priori, irrespective of the clustering methods, the final partition of data requires some kind of evaluation in most applications.

### 2.9 Conclusions

This chapter has introduced the clustering process, the various clustering algorithms, the genetic algorithm, fuzzy logic concepts, fuzzy membership function design, the grouping problem and clustering indices. While the focus has been on clustering algorithms, the other topics included were chosen selectively. The model adopts a soft computing approach to clustering. It requires a GA aided by a fuzzy inferencing system. The model addresses the grouping problem in particular. The model is modified to generate fuzzy membership functions from numerical data on the variables. In this context, the conventional approaches to membership function design were also reviewed. The survey provides the necessary background for introduction of the soft computing model in the next chapter.