# A Survey on Soft Computing Techniques for clustering

Shweta Gupta , Ms Jasmine Jha
*Department of Computer Science & Engineering ,L.J. Institute of Engineering & Technology*

***Abstract*- In recent years, data mining has a zealous attention in the information industry as well as society because of broad availability of huge amounts of data and the near need for turning such data into useful knowledge and information. Handle this huge amount of data we partition this data into small groups. This grouping is done using clustering. Clustering is a process of putting similar data into groups. A clustering algorithm partitions a data set into several groups such that the similarity within a group is larger than among groups. Soft Computing is the modern approach for constructing a computationally intelligent system. The upmost goal of soft computing is to imitate human mind as closely as possible. Soft computing is the blend of methodologies designed to solve the real-world problems, which are not solved or too tough to solve mathematically. This paper presents a survey of soft computing techniques designed for clustering tasks.**

***Index Term*- Clustering, Soft Computing, Fuzzy Logic, Genetic Algorithm, partitioning algorithms, hierarchical algorithms.**

## I. INTRODUCTION

CLUSTERING means the act of partitioning an unlabeled data set into groups of similar objects. Each group, called a "cluster," consists of objects that are similar between themselves and dissimilar to objects of other groups. In the past few decades, cluster analysis has played a central role in a variety of fields, ranging from engineering (e.g., machine learning, artificial intelligence, pattern recognition, mechanical engineering, and electrical engineering), computer sciences (e.g., web mining, spatial database analysis, textual document collection, and image segmentation), and life and medical sciences (e.g., genetics, biology, microbiology, paleontology, psychiatry, and pathology) to earth sciences (e.g., geography, geology, and remote sensing), social sciences (e.g., sociology, psychology, archeology, and education), and economics (e.g., marketing and business) [16]-[23].

Data clustering algorithms can be hierarchical or partitional [24], [25]. Within each of the types, there exists a wealth of subtypes and different algorithms for finding the clusters. In hierarchical clustering, the output is a tree showing a sequence of clustering, with each cluster being a partition of the data set [25].Hierarchical algorithms can be agglomerative (bottom-up) or divisive (top-down). Agglomerative algorithms begin with each element as a separate cluster and merge them in successively larger clusters. Divisive algorithms begin with the whole set and proceed to divide it into successively smaller clusters. Hierarchical algorithms have two basic advantages [24]. First, the number of classes need not be specified a priori, and second, they are independent of the initial conditions. However, the main drawback of hierarchical clustering techniques is that they are static; that is, data points assigned to a cluster cannot move to another cluster. In addition to that, they may fail to separate overlapping clusters due to lack of information about the global shape or size of the clusters [26]. Partitioning clustering algorithms, on the other hand, attempt to decompose the data set directly into a set of disjoint clusters. They try to optimize certain criteria. The criterion function may emphasize the local structure of the data, such as by assigning clusters to peaks in the probability density function, or the global structure. Typically, the global criteria involve minimizing some measure of dissimilarity in the samples within each cluster while maximizing the dissimilarity of different clusters. The advantages of the hierarchical algorithms are the disadvantages of the partitioning algorithms, and vice versa. An extensive survey of various clustering techniques can be found in [26].

Soft Computing is the modern approach for constructing a computationally intelligent system. The upmost goal of soft computing is to imitate human mind as closely as possible. Soft computing is the blend of methodologies designed to solve the real-world problems, which are not solved or too tough to solve mathematically. Currently, soft computing techniques are identified as attractive alternatives to the standard, well-established "hard computing" methods. Hard computing methods are often inconvenient for real-world problems. They always need an exactly stated systematic model and often a lot of computational time. Soft computing consists of several techniques, the very important being neural networks, fuzzy logic and genetic algorithms. The value of soft computing techniques lies in the fact that they are complementary, not competitive. In several cases a problem can be solved by using fuzzy logic, neural network, and genetic algorithm in combination rather than exclusively.

## II. SOFT COMPUTING

Soft Computing is the modern approach for constructing a computationally intelligent system. The upmost goal of soft computing is to imitate human

mind as closely as possible. Soft computing is the blend of methodologies designed to solve the real-world problems, which are not solved or too tough to solve mathematically. Currently, soft computing techniques are identified as attractive alternatives to the standard, well-established "hard computing" methods. Hard computing methods are often inconvenient for real-world problems. They always need an exactly stated systematic model and often a lot of computational time. Soft computing consists of several techniques, the very important being neural networks, fuzzy logic and genetic algorithms. The value of soft computing techniques lies in the fact that they are complementary, not competitive. In several cases a problem can be solved by using fuzzy logic, neural network, and genetic algorithm in combination rather than exclusively.

Soft computing is a term applied to a field within computer science which is characterized by the use of inexact solutions to computationally hard tasks such as the solution of NP-complete problems, for which there is no known algorithm that can compute an exact solution in polynomial time [11]. The idea of Soft Computing was initiated by Lotfi A. Zadeh. Soft Computing is an emerging approach to computing which parallel the remarkable ability of human mind to reason and learn in an environment of uncertainty (doubt) and imprecision.

Zadeh defines SC into one multidisciplinary system as the fusion of the fields of Fuzzy Logic, Neuro Computing, Genetic Computing and Probabilistic Computing. Merging of methodologies designed to model and enable solutions to actual world problems, which are not modeled or too complex to model mathematically. They are composed of two features: "adaptively" & "knowledge". Soft computing is foundation of conceptual intelligence in machines.

Soft computing different from Hard computing (conventional computing) in that it is tolerant of following:

- Imprecision
- Uncertainty
- Partial truth
- Approximation

The role model for soft computing is human mind. Soft Computing is a term used in computer science to refer to problems in computer science whose solutions are unpredictable, uncertain and between 0 and 1[11]. Soft computing deals with imprecision, uncertainty, partial truth, and approximation to achieve practicability, robustness and low solution cost. Recent trends tend to involve evolutionary and swarm intelligence based algorithms and bio-inspired computation.

The main goal is to develop intelligent machines to provide solutions to real world problems. Its aim is to exploit (develop) the tolerance for Approximation, Uncertainty, Imprecision, and Partial Truth in order to achieve close resemblance with human like decision making.

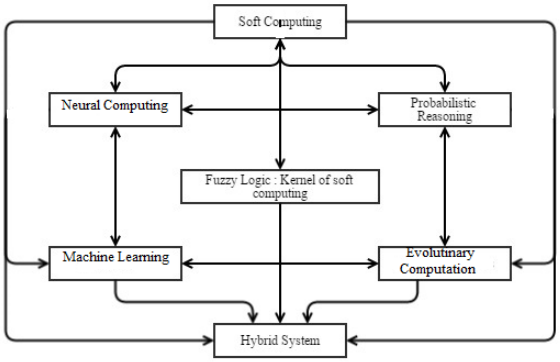


Figure 2.1 Relation between Soft computing Components

Principal components of soft computing:

- Fuzzy Logic (FL)
- Neural Computing (NC)
- Evolutionary Computation (EC)
- Machine Learning (ML)
- Probabilistic Reasoning (PR)

## III. LITERATURE SURVEY

**Evolutionary Computation**

This section serves as a Basis for the algorithmic framework developed in this Dissertation.

Nature has been continuously offering us optimization models. In the last decades, some of these models served as inspiration for the development of computational methods that aim at overcoming the increasing complexity of the problems addressed by the modern human society. In this context, this section presents the Evolutionary Computation field in a top-down approach. First, the general principles shared by the Evolutionary Computation methods are stated. Then, two of the main paradigms components of the Evolutionary Computation field - Evolutionary Algorithms and Swarm Intelligence are described. Two particular optimization methods, exponents of the two paradigms used across this thesis are detailed: the Genetic Algorithms and Particle Swarm Optimization.

General Principle

Evolutionary Computation (EC) comprises a set of soft-computing paradigms designed to solve optimization problems. In contrast with the rigid models of hard computing, these nature-inspired models provide self-adaptation mechanisms which aim at identifying and exploiting the properties of the instance of the problem being solved.

EC methods are iterative algorithms. They work with a population of candidate solutions which evolve in order to adapt to the "environment" defined by a

"fitness function". They involve a degree of randomness which classifies them as probabilistic methods. Several approximate good solutions are returned.
An important advantage over classical computational methods is their extended usability. There exist several heuristics which comply with the guidelines listed above. Most of them can be grouped in two major classes:

1. Evolutionary Algorithms (EA) and
2. Swarm Intelligence (SI) algorithms

The main differences between the two paradigms come as result of their different sources of inspiration. EA methods have roots in biological evolution while SI methods simulate the behavior of decentralized self-organized systems. The current thesis makes use of techniques of both types; therefore, the next two sections describe in detail these two paradigms.

1.Evolutionary Algorithms

Evolutionary algorithms use a vocabulary borrowed from genetics. They simulate the evolution across a sequence of generations (iterations within an iterative process) of a population (set) of candidate solutions. A candidate solution is internally represented as a string of genes and is called chromosome or individual. The position of a gene in a chromosome is called locus and all the possible values for the gene form the set of alleles of the respective gene. The internal representation (encoding) of a candidate solution in an evolutionary algorithm form the genotype; this information is processed by the evolutionary algorithm. Each chromosome corresponds to a candidate solution in the search space of the problem which represents its phenotype. A decoding function is necessary to translate the genotype into phenotype. If the search space is finite, it is desirable that this function should satisfy the bijection property in order to avoid redundancy in chromosomes encoding (which would slow down the convergence) and to ensure the coverage of the entire search space.

The population maintained by an evolutionary algorithm evolves with the aid of genetic operators that simulate the fundamental elements in genetics: mutation consists in a random perturbation of a gene while crossover aims at exchanging genetic information among several chromosomes. The chromosome subjected to a genetic operator is called parent and the resulted chromosome is called offspring.

A process called selection involving some degree of randomness selects the individuals to breed and create offsprings, mainly based on individual merit. The individual merit is measured using a fitness function which quantifies how fitted the candidate solution encoded by the chromosome is for the problem being solved. The fitness function is formulated based on the mathematical function to be optimized.

The solution returned by an evolutionary algorithm is usually the most fitted chromosome in the last generation.

**Directions in Evolutionary Algorithms**

First efforts to develop computational models of evolutionary systems date back to 1950s Bremermann; Fraser. Several distinct interpretations, which are widely used nowadays were independently developed later. The main differences between these classes of evolutionary algorithms consist in solution encoding, operators implementation and selection schemes.
*Evolutionary programming* crystallized in 1963 in the USA at San Diego University, when Lawrence J. Fogel in 1966 generated simple programs as simple finitestate machines; this technique was developed further by his son David Fogel (1992). A random mutation operator was applied on state-transition diagrams and the best chromosome was selected for survival.
*Evolutionary strategies (ES)* were introduced in 1960s when Hans-Paul Schwefel and Ingo Rechenberg, working on a problem from mechanics involving shape optimization, designed a new optimization technique because existing mathematical methods were unable to provide a solution. The first ES algorithm was initially proposed by Schwefel in 1965 and developed further by Rechenberg in 1973. Their idea is known as Rechenberg's conjecture, and states the fundamental justification for the use of evolutionary techniques: "Natural evolution is, or comprises, a very efficient optimization process, which, by simulation, can conduct to solving difficult optimization processes". Their method was designed to solve optimization problems with continuous variables; it used one candidate solution and applied random mutations followed by the selection of the fittest. Evolutionary strategies were later strongly promoted by Thomas Back in 1996 that incorporated the idea of population of solutions.
*Genetic algorithms* were developed by John Henry Holland in 1973 after years of study of the idea of simulating the natural evolution. These algorithms model the genetic inheritance and the Darwinian competition for survival.
*Genetic Programming (GP)* is a specialized form of a genetic algorithm. The specialization consists in manipulating a very specific type of encoding and, consequently, in using modified versions of the genetic operators. GP was introduced by Koza in 1992 in an attempt to perform automatic

programming. GP manipulates directly phenotypes, which are computer programs (hierarchical structures) expressed as trees. It is currently intensively used to solve symbolic regression problems.
*Differential evolution* was introduced by Storn and Price in 1997. It is a more recent class of evolutionary algorithms whose operators are specifically designed for numerical optimization.

2.Swarm Intelligence

Swarm Intelligence (SI) is a computational paradigm inspired from the collective behavior in auto-organized decentralized systems. It stipulates that problem solving can emerge at the level of a collection of agents which are not aware of the problem itself, but collective interactions lead to the solution. Swarm Intelligence systems are typically made up of a population of simple autonomous agents interacting locally with one another and with their environment. Although there is no centralized control, the local interactions between agents lead to the emergence of global behavior. Examples of systems like this can be found in nature, including ant colonies, bird flocking, animal herding, bacteria molding and fish schooling.

The most successful SI techniques are Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO). In ACO [15] artificial ants build solutions walking in the graph of the problem and (simulating real ants) leaving artificial pheromone so that other ants will be able to build better solutions. ACO was successfully applied to an impressive number of optimization problems. PSO is an optimization method initially designed for continuous optimization; however, it was further adapted to solve various combinatorial problems.

## IV. CONCLUSION

This paper presents a survey on evolutionary computation for clustering. It tries to reflect the profile of this area by focusing more on those subjects that have been given more importance in the literature. We also briefly discuss the soft computing techniques because the evolutionary algorithm is a part of soft computing.

## REFERENCES

[1] Jiawei Han, Micheline Kamber, Simon Fraser University, A book on "Data Mining: Concepts and Techniques",Academic Press, Morgan Kaufmann Publishers, 2006.

[2] B.N. Lakshmi,G.H. Raghunandhan, "A Conceptual Overview of Data Mining", National Conference on Innovations in Emerging Technology, India, February 17-18, IEEE 2011, ISSN: 978-1-61284-810-5.

[3] Daxin Jiang, Chun Tang, and Aidong Zhang, "Cluster Analysis for Gene Expression Data: A Survey", IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 16, NO. 11, NOVEMBER 2004.

[4] J. Kleinberg. An impossibility theorem for clustering. In S. Becker, S. Thrun, and K. Obermayer, editors, Advances in Neural Information Processing Systems, pages 446–453. MIT Press, 2002.ISBN 0-262-02550-7.

[5] Richard O. Duda, Peter E. Hart, and David G. Stork. Pattern classification. Wiley, 2 edition, November 2001. ISBN 0471056693.

[6] Rui Xu and II Wunsch, D. Survey of clustering algorithms. Neural Networks, IEEE Transactions on, 16(3):645 –678, May 2005. ISSN 1045-9227. doi: 10.1109/TNN. 2005.845141.

[7] S. Guha. Cure: an efficient clustering algorithm for large databases. Information Systems, 26(1):35–58, March 2001. ISSN 03064379.

[8] George Karypis, Eui-Hong (Sam) Han, and Vipin Kumar. Chameleon: Hierarchical clustering using dynamic modeling. Computer, 32:68–75, 1999. ISSN 0018-9162.

[9] Martin Ester, Hans-Peter Kriegel, Jrg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proc. of 2nd International Conference on Knowledge Discovery and, pages 226–231, 1996.

[10] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and J?rg Sander. Optics: Ordering points to identify the clustering structure. ACM SIGMOD Record, 28(2): 49–60, 1999. ISSN 0163-5808.

[11] Leonard Kaufman and Peter J. Rousseeuw. Finding Groups in Data: An Introduction to Cluster Analysis (Wiley Series in Probability and Statistics). Wiley-Interscience, 2005. ISBN 0471735787.

[12] Forgy. Cluster analysis of multivariate data: Efficiency versus interpretability of classification. Biometrics, 21(3):768–769, 1965.

[13] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, and Prabhakar Raghavan. Automatic subspace clustering of high dimensional data. Data Min. Knowl. Discov., 11:5–33, July 2005. ISSN 1384-5810.

[14] Ada, Wai-Chee Fu, Cheng Chun-Hung, and Cheng Chun-Hung. Entropy-based Subspace Clustering for Mining Numerical Data. 1999.

[15] Marco Dorigo and Thomas Stu¨tzle. Ant Colony Optimization. Bradford Company, Scituate, MA, USA, 2004. ISBN 0262042193.

[16] I. E. Evangelou, D. G. Hadjimitsis, A. A. Lazakidou, and C. Clayton, "Data mining and knowledge discovery in complex image data using artificial neural networks," in Proc. Workshop Complex Reason. Geogr. Data, Paphos, Cyprus, 2001.

[17] T. Lillesand and R. Keifer, Remote Sensing and Image Interpretation. Hoboken, NJ: Wiley, 1994.

[18] H. C. Andrews, Introduction to Mathematical Techniques in Pattern Recognition. New York: Wiley, 1972.

[4] M. R. Rao, "Cluster analysis and mathematical programming," J. Amer. Stat. Assoc., vol. 66, no. 335, pp. 622–626, Sep. 1971.

[5] R. O. Duda and P. E. Hart,Pattern Classification and Scene Analysis. Hoboken, NJ: Wiley, 1973.

[19] K.Fukunaga,Introduction to Statistical Pattern Recognition. NewYork: Academic, 1990.

[20] B. S. Everitt, Cluster Analysis, 3rd ed. New York: Halsted, 1993.

[21] J. A. Hartigan, Clustering Algorithms. New York: Wiley, 1975.

[22] H. Frigui and R. Krishnapuram, "A robust competitive clustering algorithm with applications in computer vision," IEEE Trans. Pattern Anal. Mach. Intell., vol. 21, no. 5, pp. 450–465, May 1999.

[23] Y. Leung, J. Zhang, and Z. Xu, "Clustering by scale-space filtering," IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 12, pp. 1396–1410, Dec. 2000.

[24] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," ACM Comput. Surv., vol. 31, no. 3, pp. 264–323, Sep. 1999.

[25] R. Xu, D. Wunsch II, "Survey of Clustering Algorithms", IEEE Trans. on Neural Networks, Vol. 16, pp. 645-678, 2005.
[13] J.V. de Oliveira, W. Pedrycz, Advances in Fuzzy Clustering and its Applications, Wiley, 2007.