Swarm Intelligence Algorithms for Data Clustering

Ajith Abraham[1], Swagatam Das[2], and Sandip Roy[3]

[1] Center of Excellence for Quantifiable Quality of Service (Q2S), Norwegian University of Science and Technology, Trondheim, Norway
ajith.abraham@ieee.org
[2] Department of Electronics and Telecommunication Engineering, Jadavpur University, Kolkata 700032, India.
[3] Department of Computer Science and Engineering, Asansol Engineering College, Asansol-713304, India.

Summary. Clustering aims at representing large datasets by a fewer number of prototypes or clusters. It brings simplicity in modeling data and thus plays a central role in the process of knowledge discovery and data mining. Data mining tasks, in these days, require fast and accurate partitioning of huge datasets, which may come with a variety of attributes or features. This, in turn, imposes severe computational requirements on the relevant clustering techniques. A family of bio-inspired algorithms, well-known as Swarm Intelligence (SI) has recently emerged that meets these requirements and has successfully been applied to a number of real world clustering problems. This chapter explores the role of SI in clustering different kinds of datasets. It finally describes a new SI technique for partitioning any dataset into an optimal number of groups through one run of optimization. Computer simulations undertaken in this research have also been provided to demonstrate the effectiveness of the proposed algorithm.

1 Introduction

Clustering means the act of partitioning an unlabeled dataset 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 (machine learning, artificial intelligence, pattern recognition, mechanical engineering, electrical engineering), computer sciences (web mining, spatial database analysis, textual document collection, image segmentation), life and medical sciences (genetics, biology, microbiology, paleontology, psychiatry, pathology), to earth sciences (geography. geology, remote sensing), social sciences (sociology, psychology, archeology, education),

and economics (marketing, business) (Evangelou *et al.*, 2001, Lillesand and Keifer, 1994, Rao, 1971, Duda and Hart, 1973, Fukunaga, 1990, Everitt, 1993).

From a machine learning perspective, clusters correspond to the hidden patterns in data, the search for clusters is a kind of unsupervised learning, and the resulting system represents a *data concept*. The problem of data clustering has been approached from diverse fields of knowledge like statistics (multivariate analysis) (Forgy, 1965), graph theory (Zahn, 1971), expectation maximization algorithms (Mitchell, 1997), artificial neural networks (Mao and Jain, 1995, Pal *et al.*, 1993, Kohonen, 1995), evolutionary computing (Falkenauer, 1998, Paterlini and Minerva, 2003) and so on. Researchers all over the globe are coming up with new algorithms, on a regular basis, to meet the increasing complexity of vast real-world datasets. A comprehensive review of the state-of-the-art clustering methods can be found in (Xu and Wunsch, 2005) and (Rokach and Maimon, 2005).

Data mining is a powerful new technology, which aims at the extraction of hidden predictive information from large databases. Data mining tools predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions. The process of knowledge discovery from databases necessitates fast and automatic clustering of very large datasets with several attributes of different types (Mitra *et al.*, 2002). This poses a severe challenge before the classical clustering techniques. Recently a family of nature inspired algorithms, known as *Swarm Intelligence* (SI), has attracted several researchers from the field of pattern recognition and clustering. Clustering techniques based on the SI tools have reportedly outperformed many classical methods of partitioning a complex real world dataset.

Swarm Intelligence is a relatively new interdisciplinary field of research, which has gained huge popularity in these days. Algorithms belonging to the domain, draw inspiration from the collective intelligence emerging from the behavior of a group of social insects (like bees, termites and wasps). When acting as a community, these insects even with very limited individual capability can jointly (cooperatively) perform many complex tasks necessary for their survival. Problems like finding and storing foods, selecting and picking up materials for future usage require a detailed planning, and are solved by insect colonies without any kind of supervisor or controller. An example of particularly successful research direction in swarm intelligence is Ant Colony Optimization (ACO) (Dorigo *et al.*, 1996, Dorigo and Gambardella, 1997), which focuses on discrete optimization problems, and has been applied successfully to a large number of NP hard discrete optimization problems including the traveling salesman, the quadratic assignment, scheduling, vehicle routing, etc., as well as to routing in telecommunication networks. Particle Swarm Optimization (PSO) (Kennedy and Eberhart, 1995) is another very popular SI algorithm for global optimization over continuous search spaces. Since its advent in 1995, PSO has attracted the attention of several researchers all over the world resulting into a huge number of variants of the basic algorithm as well as many parameter automation strategies.

In this Chapter, we explore the applicability of these bio-inspired approaches to the development of self-organizing, evolving, adaptive and autonomous clustering techniques, which will meet the requirements of next-generation data mining systems, such as diversity, scalability, robustness, and resilience. The next section of the chapter provides an overview of the SI paradigm with a special emphasis on two SI algorithms well-known as Particle Swarm Optimization (PSO) and Ant Colony Systems (ACS). Section 3 outlines the data clustering problem and briefly reviews the present state of the art in this field. Section 4 describes the use of the SI algorithms in both crisp and fuzzy clustering of real world datasets. A new automatic clustering algorithm, based on PSO, has been outlined in this Section. The algorithm requires no previous knowledge of the dataset to be partitioned, and can determine the optimal number of classes dynamically. The new method has been compared with two well-known, classical fuzzy clustering algorithms. The Chapter is concluded in Section 5 with possible directions for future research.

2 An Introduction to Swarm Intelligence

The behavior of a single ant, bee, termite and wasp often is too simple, but their collective and social behavior is of paramount significance. A look at National Geographic TV Channel reveals that advanced mammals including lions also enjoy social lives, perhaps for their self-existence at old age and in particular when they are wounded. The collective and social behavior of living creatures motivated researchers to undertake the study of today what is known as *Swarm Intelligence*. Historically, the phrase Swarm Intelligence (SI) was coined by Beny and Wang in late 1980s (Beni and Wang, 1989) in the context of cellular robotics. A group of researchers in different parts of the world started working almost at the same time to study the versatile behavior of different living creatures and especially the social insects. The efforts to mimic such behaviors through computer simulation finally resulted into the fascinating field of SI. SI systems are typically made up of a population of simple agents (an entity capable of performing/executing certain operations) interacting locally with one another and with their environment. Although there is normally no centralized control structure dictating how individual agents should behave, local interactions between such agents often lead to the emergence of global behavior. Many biological creatures such as fish schools and bird flocks clearly display structural order, with the behavior of the organisms so integrated that even though they may change shape and direction, they appear to move as a single coherent entity (Couzin *et al.*, 2002). The main properties of the collective behavior can be pointed out as follows and is summarized in Figure 1.

Homogeneity: every bird in flock has the same behavioral model. The flock moves without a leader, even though temporary leaders seem to appear.

Locality: its nearest flock-mates only influence the motion of each bird. Vision is considered to be the most important senses for flock organization.
Collision Avoidance: avoid colliding with nearby flock mates.
Velocity Matching: attempt to match velocity with nearby flock mates.
Flock Centering: attempt to stay close to nearby flock mates

Individuals attempt to maintain a minimum distance between themselves and others at all times. This rule is given the highest priority and corresponds to a frequently observed behavior of animals in nature (Krause and Ruxton, 2002). If individuals are not performing an avoidance maneuver they tend to be attracted towards other individuals (to avoid being isolated) and to align themselves with neighbors (Partridge and Pitcher, 1980, Partridge, 1982).

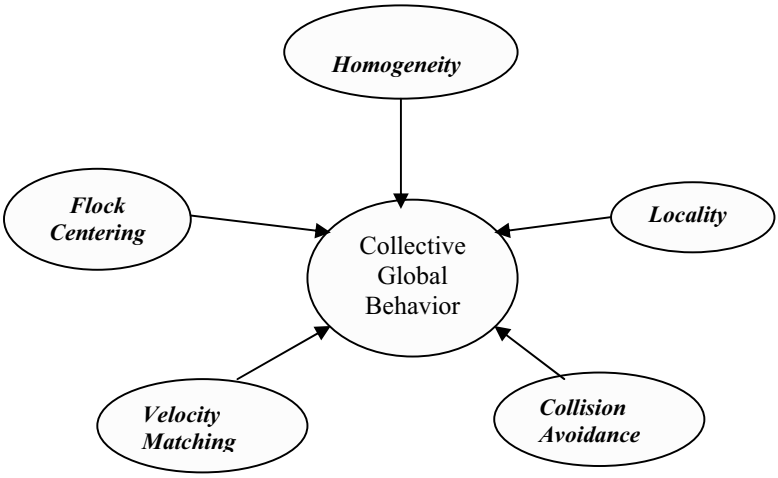

Fig. 1. Main traits of collective behavior

Couzin *et al.* identified four collective dynamical behaviors (Couzin *et al.*, 2002) as illustrated in Figure 2:

Swarm: an aggregate with cohesion, but a low level of polarization (parallel alignment) among members
Torus: individuals perpetually rotate around an empty core (milling). The direction of rotation is random.
Dynamic parallel group: the individuals are polarized and move as a coherent group, but individuals can move throughout the group and density and group form can fluctuate (Partridge and Pitcher, 1980, Major and Dill, 1978).
Highly parallel group: much more static in terms of exchange of spatial positions within the group than the dynamic parallel group and the variation in density and form is minimal.

As mentioned in (Grosan *et al.*, 2006) at a high-level, a swarm can be viewed as a group of agents cooperating to achieve some purposeful behavior and achieve some goal (Abraham *et al.*, 2006). This collective intelligence seems to emerge from what are often large groups:

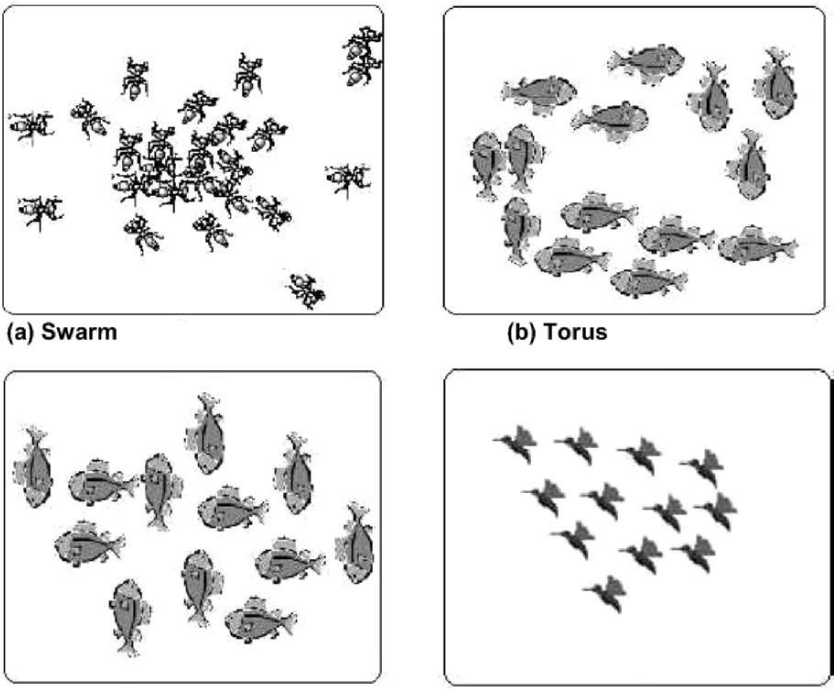

Fig. 2. Different models of collective behavior (Grosan *et al.*, 2006)

According to Milonas, five basic principles define the SI paradigm (Milonas, 1994). First is the the proximity principle: the swarm should be able to carry out simple space and time computations. Second is the quality principle: the swarm should be able to respond to quality factors in the environment. Third is the principle of diverse response: the swarm should not commit its activities along excessively narrow channels. Fourth is the principle of stability: the swarm should not change its mode of behavior every time the environment changes. Fifth is the principle of adaptability: the swarm must be able to change behavior mote when it is worth the computational price. Note that principles four and five are the opposite sides of the same coin. Below we discuss in details two algorithms from SI domain, which have gained wide popularity in a relatively short span of time.

2.1 The Ant Colony Systems

The basic idea of a real ant system is illustrated in Figure 4. In the left picture, the ants move in a straight line to the food. The middle picture illustrates the situation soon after an obstacle is inserted between the nest and the food. To avoid the obstacle, initially each ant chooses to turn left or right at random. Let us assume that ants move at the same speed depositing pheromone in the trail uniformly. However, the ants that, by chance, choose to turn left will reach the food sooner, whereas the ants that go around the obstacle turning right will follow a longer path, and so will take longer time to circumvent the obstacle. As a result, pheromone accumulates faster in the shorter path around the obstacle. Since ants prefer to follow trails with larger amounts of pheromone, eventually all the ants converge to the shorter path around the obstacle, as shown in Figure 3.

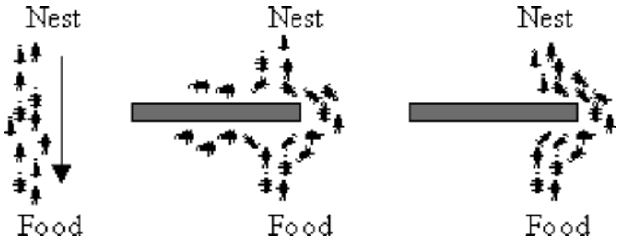

Fig. 3. Illustrating the behavior of real ant movements.

An artificial Ant Colony System (ACS) is an agent-based system, which simulates the natural behavior of ants and develops mechanisms of cooperation and learning. ACS was proposed by Dorigo *et al.* (Dorigo and Gambardella, 1997) as a new heuristic to solve combinatorial optimization problems. This new heuristic, called Ant Colony Optimization (ACO) has been found to be both robust and versatile in handling a wide range of combinatorial optimization problems.

The main idea of ACO is to model a problem as the search for a minimum cost path in a graph. Artificial ants as if walk on this graph, looking for cheaper paths. Each ant has a rather simple behavior capable of finding relatively costlier paths. Cheaper paths are found as the emergent result of the global cooperation among ants in the colony. The behavior of artificial ants is inspired from real ants: they lay pheromone trails (obviously in a mathematical form) on the graph edges and choose their path with respect to probabilities that depend on pheromone trails. These pheromone trails progressively decrease by evaporation. In addition, artificial ants have some extra features not seen in their counterpart in real ants. In particular, they live in a discrete world (a graph) and their moves consist of transitions from nodes to nodes.

Below we illustrate the use of ACO in finding the optimal tour in the classical Traveling Salesman Problem (TSP). Given a set of n cities and a set of distances between them, the problem is to determine a minimum traversal of the cities and return to the home-station at the end. It is indeed important to note that the traversal should in no way include a city more than once. Let $r \quad (C_x, C_y)$be a measure of cost for traversal from city C_x to C_y. Naturally, the total cost of traversing n cities indexed by $i_1, i_2, i_3, \ldots, i_n$ in order is given by the following expression:

$$Cost(i_1, i_2, ..., i_n) = \sum_{j=1}^{n-1} r(Ci_j, Ci_{j+1}) + r(Ci_n, Ci_1) \tag{1}$$

The ACO algorithm is employed to find an optimal order of traversal of the cities. Let τ be a mathematical entity modeling the pheromone and $\eta_{ij} =$ 1/r (i , j) is a local heuristic. Also let allowed$_k$(t) be the set of cities that are yet to be visited by ant k located in city$i$. Then according to the classical ant system (Everitt, 1993) the probability that ant k in city i visits city j is given by:

$$p_{ij}^k(t) = \begin{array}{ll} \frac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_{h \in allowed_k(t)} [\tau_{ih}(t)]^\alpha [\eta_{ih}]^\beta} & if\, h \in allowed_k(t) \\ 0 & otherwise \end{array} \tag{2}$$

In Equation 2 shorter edges with greater amount of pheromone are favored by multiplying the pheromone on edge $(i, j\ )$ by the corresponding heuristic value $\eta(i, j\ )$. Parameters α (¿ 0) and β (¿ 0) determine the relative importance of pheromone versus cost. Now in ant system, pheromone trails are updated as follows. Let D_k be the length of the tour performed by ant k, $\Delta\tau_k$ (i , j)= 1/D$_k$ if $(i, j) \quad \in$ tour done by ant $k$and = 0 otherwise and finally let ρ $\in$ [0,1] be a pheromone decay parameter which takes care of the occasional evaporation of the pheromone from the visited edges. Then once all ants have built their tours, pheromone is updated on all the ages as,

$$\tau(i, j) = (1 - \rho).\tau(i, j) + \sum_{k=1}^{m} \Delta\tau_k(i, j) \tag{3}$$

From Equation (3), we can guess that pheromone updating attempts to accumulate greater amount of pheromone to shorter tours (which corresponds to high value of the second term in (3) so as to compensate for any loss of pheromone due to the first term). This conceptually resembles a reinforcement-learning scheme, where better solutions receive a higher reinforcement.

The ACO differs from the classical ant system in the sense that here the pheromone trails are updated in two ways. Firstly, when ants construct a tour they locally change the amount of pheromone on the visited edges by a local

updating rule. Now if we let γ to be a decay parameter and $\Delta\tau(\mathrm{i, j}) = \tau_0$ such that τ_0 is the initial pheromone level, then the local rule may be stated as,

$$\tau(i,j) = (1-\gamma).\tau(i,j) + \gamma.\Delta\tau(i,j) \tag{4}$$

Secondly, after all the ants have built their individual tours, a global updating rule is applied to modify the pheromone level on the edges that belong to the best ant tour found so far. If κ be the usual pheromone evaporation constant, D_{gb} be the length of the globally best tour from the beginning of the trial and

$\Delta\tau^{/}(\mathrm{i\ ,\ j}) = 1/\ \mathrm{D}_{gb}$ only when the edge (i,j) belongs to global-best-tour and zero otherwise, then we may express the global rule as follows:

$$\tau(i,j) = (1-\kappa).\tau(i,j) + \kappa.\Delta\tau^{/}(i,j) \tag{5}$$

The main steps of ACO algorithm are presented in Algorithm 1.

Algorithm 1: Procedure ACO

1: Initialize pheromone trails;
2: **repeat** {at this stage each loop is called an iteration}
3: Each ant is positioned on a starting node
4: **repeat** {at this level each loop is called a step}
5: Each ant applies a *state transition rule like rule (2)* to incrementally build a solution and a *local pheromone-updating rule like rule (4);*
6: **until** all ants have built a complete solution
7: global pheromone-updating rule like rule (5) is applied.
8: **until** terminating condition is reached

2.2 The Particle Swarm Optimization (PSO)

The concept of Particle Swarms, although initially introduced for simulating human social behaviors, has become very popular these days as an efficient search and optimization technique. The Particle Swarm Optimization (PSO) (Kennedy and Eberhart, 1995,Kennedy *et al.*, 2001), as it is called now, does not require any gradient information of the function to be optimized, uses only primitive mathematical operators and is conceptually very simple.

In PSO, a population of conceptual 'particles' is initialized with random positions X_i and velocities V_i, and a function, f, is evaluated, using the particle's positional coordinates as input values. In an n-dimensional search space, $\mathrm{X}_i = (\mathrm{x}_{i1}, \mathrm{x}_{i2}, \mathrm{x}_{i3},...,\mathrm{x}_{in})$ and $\mathrm{V}_i = (\mathrm{v}_{i1}, \mathrm{v}_{i2}, \mathrm{v}_{i3},...,\mathrm{v}_{in})$. Positions and velocities are adjusted, and the function is evaluated with the new coordinates at each time-step. The basic update equations for the d-th dimension of the i-th particle in PSO may be given as

$$V_{id}(t+1) = \omega.V_{id}(t) + C_1.\varphi_1.(P_{lid} - X_{id}(t)) + C_2.\varphi_2.(P_{gd} - X_{id}(t))$$
$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1) \quad (6)$$

The variables ϕ_1 and ϕ_2 are random positive numbers, drawn from a uniform distribution and defined by an upper limit $\phi_{max,}$which is a parameter of the system. C_1 and C_2 are called acceleration constants whereas ω is called inertia weight. P_{li} is the local best solution found so far by the i-th particle, while P_g represents the positional coordinates of the fittest particle found so far in the entire community. Once the iterations are terminated, most of the particles are expected to converge to a small radius surrounding the global optima of the search space. The velocity updating scheme has been illustrated in Figure 4 with a humanoid particle.

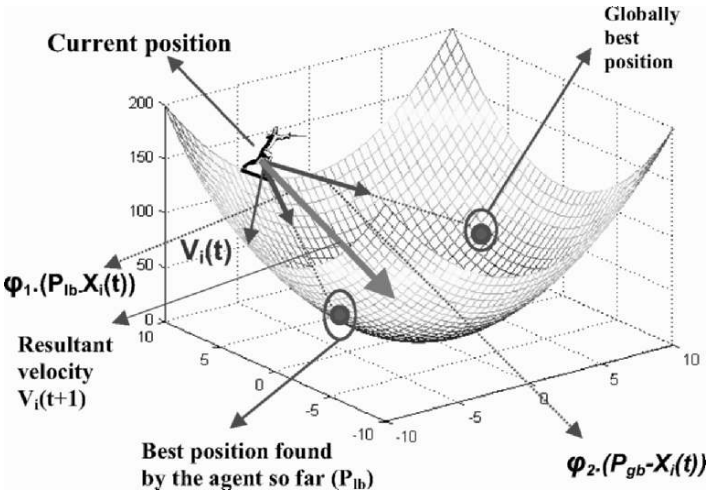

Fig. 4. Illustrating the velocity updating scheme of basic PSO

A pseudo code for the PSO algorithm is presented in Algorithm 2.

3 Data Clustering – An Overview

In this section, we first provide a brief and formal description of the clustering problem. We then discuss a few major classical clustering techniques.

3.1 Problem Definition

A *pattern* is a physical or abstract structure of objects. It is distinguished from others by a collective set of attributes called *features*, which together represent

Algorithm 2: The PSO Algorithm

Input: Randomly initialized position and velocity of the particles: $\mathbf{X}_i(0)$ and $\mathbf{V}_i(0)$
Output: Position of the approximate global optima $\mathbf{X}^*$

```
while terminating condition is not reached do
  for i = 1 to numberofparticles do
    Evaluate the fitness: =f(X_i(t));
    Update P(t)andg(t);
    Adapt velocity of the particle using Equation 3;
    Update the position of the particle;
  end for
end while
```

a pattern (Konar, 2005). Let P = {P_1, P_2... P_n} be a set of n patterns or data points, each having d features. These patterns can also be represented by a profile data matrix $\mathbf{X}_{n\times d}$ having n d-dimensional row vectors. The i-th row vector $\mathbf{X}_i$ characterizes the i-th object from the set P and each element $X_{i,j}$ in $\mathbf{X}_i$ corresponds to the j-th real value feature ($j = 1, 2, ..., d$) of the i-th pattern (i =1,2,...., n). Given such an $\mathbf{X}_{n\times d,}a$ partitional clustering algorithm tries to find a partition C = {C_1, C_2,....., C_K} of K classes, such that the similarity of the patterns in the same cluster is maximum and patterns from different clusters differ as far as possible. The partitions should maintain the following properties:

1. Each cluster should have at least one pattern assigned i.e. $C_i \neq \Phi \forall i \in \{1, 2, ..., K\}$.
2. Two different clusters should have no pattern in common. i.e. $C_i \cap C_j = \Phi, \forall i \neq j$ and $i, j \in \{1, 2, ..., K\}$. This property is required for crisp (hard) clustering. In Fuzzy clustering this property doesn't exist.
3. Each pattern should definitely be attached to a cluster i.e. $\bigcup_{i=1}^{K} C_i = P$.

Since the given dataset can be partitioned in a number of ways maintaining all of the above properties, a fitness function (some measure of the adequacy of the partitioning) must be defined. The problem then turns out to be one of finding a partition $\mathbf{C}^*$ of optimal or near-optimal adequacy as compared to all other feasible solutions $\mathbf{C}$ = { C^1, C^2,....., $C^{N(n,K)}$} where,

$$N(n, K) = \frac{1}{K!} \sum_{i=1}^{K} (-1)^i \binom{K}{i}^i (K - i)^i \tag{7}$$

is the number of feasible partitions. This is same as,

$$\begin{matrix} Optimizef(X_{\mathrm{n\times d}}, C) \\ C \end{matrix} \tag{8}$$

where C is a single partition from the set $\mathbf{C}$ and f is a statistical-mathematical function that quantifies the goodness of a partition on the basis of the similarity measure of the patterns. Defining an appropriate similarity measure plays fundamental role in clustering (Jain *et al.*, 1999). The most popular way to evaluate similarity between two patterns amounts to the use of *distance measure*. The most widely used distance measure is the Euclidean distance, which between any two d-dimensional patterns $\mathbf{X}_i$ and $\mathbf{X}_j$ is given by,

$$d(\mathbf{X}_i, \mathbf{X}_j) = \sqrt{\sum_{p=1}^{d} (X_{i,p} - X_{j,p})^2} = \|\mathbf{X}_i - \mathbf{X}_j\| \tag{9}$$

It has been shown in (Brucker, 1978) that the clustering problem is NP-hard when the number of clusters exceeds 3.

3.2 The Classical Clustering Algorithms

Data clustering is broadly based on two approaches: *hierarchical* and *partitional* (Frigui and Krishnapuram, 1999, Leung *et al.*, 2000). 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 (Leung *et al.*, 2000). 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 (Frigui and Krishnapuram, 1999). Firstly, the number of classes need not be specified a priori and secondly, they are independent of the initial conditions. However, the main drawback of hierarchical clustering techniques is they are static, i.e. data-points assigned to a cluster can not 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 (Jain *et al.*, 1999).

Partitional 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, 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 partitional algorithms and vice versa. An extensive survey of various clustering techniques can be found in (Jain *et al.*, 1999). The focus of this chapter is on the partitional clustering algorithms.

Clustering can also be performed in two different modes: crisp and fuzzy. In crisp clustering, the clusters are disjoint and non-overlapping in nature.

Any pattern may belong to one and only one class in this case. In case of fuzzy clustering, a pattern may belong to all the classes with a certain fuzzy membership grade (Jain *et al.*, 1999).

The most widely used iterative K-means algorithm (MacQueen, 1967) for partitional clustering aims at minimizing the ICS (Intra-Cluster Spread) which for K cluster centers can be defined as

$$ICS(C_1, C_2, ..., C_K) = \sum_{i=1}^{K} \sum_{\mathbf{X}_i \in C_i} \|\mathbf{X}_i - \mathbf{m}_i\|^2 \tag{10}$$

The K-means (or hard c-means) algorithm starts with K cluster-centroids (these centroids are initially selected randomly or derived from some a priori information). Each pattern in the data set is then assigned to the closest cluster-centre. Centroids are updated by using the mean of the associated patterns. The process is repeated until some stopping criterion is met.

In the c-medoids algorithm (Kaufman and Rousseeuw, 1990), on the other hand, each cluster is represented by one of the representative objects in the cluster located near the center. Partitioning around medoids (PAM) (Kaufman and Rousseeuw, 1990) starts from an initial set of medoids, and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering. Although PAM works effectively for small data, it does not scale well for large datasets. Clustering large applications based on randomized search (CLARANS) (Ng and Han, 1994), using randomized sampling, is capable of dealing with the associated scalability issue.

The fuzzy c-means (FCM) (Bezdek, 1981) seems to be the most popular algorithm in the field of fuzzy clustering. In the classical FCM algorithm, a *within cluster sum* function J_m is minimized to evolve the proper cluster centers:

$$J_m = \sum_{j=1}^{n} \sum_{i=1}^{c} (u_{ij})^m \|\mathbf{X}_j - \mathbf{V}_i\|^2 \tag{11}$$

where $\mathbf{V}_i$ is the i-th cluster center, $\mathbf{X}_j$ is the j-th d-dimensional data vector and || . || is an inner product-induced norm in d dimensions. Given *c* classes, we can determine their cluster centers $\mathbf{V}_i$ for i=1 to c by means of the following expression:

$$\mathbf{V}_i = \frac{\sum_{j=1}^{n} (u_{ij})^m \mathbf{X}_j}{\sum_{j=1}^{n} (u_{ij})^m} \tag{12}$$

Here m (m¿1) is any real number that influences the membership grade. Now differentiating the performance criterion with respect to $\mathbf{V}_i$ (treating u_{ij} as constants) and with respect to u_{ij} (treating $\mathbf{V}_i$ as constants) and setting them to zero the following relation can be obtained:

$$u_{ij} = \left[\sum_{k=1}^{c} \left(\frac{\|\mathbf{X}_j - \mathbf{V}_i\|^2}{\|\mathbf{X} - \mathbf{V}_i\|^2} \right)^{1/(m-1)} \right]^{-1} \tag{13}$$

Several modifications of the classical FCM algorithm can be found in (Hall *et al.*, 1999,Gath and Geva, 1989,Bensaid *et al.*, 1996,Clark *et al.*, 1994,Ahmed *et al.*, 2002, Wang *et al.*, 2004).

3.3 Relevance of SI Algorithms in Clustering

From the discussion of the previous section, we see that the SI algorithms are mainly stochastic search and optimization techniques, guided by the principles of collective behaviour and self organization of insect swarms. They are efficient, adaptive and robust search methods producing near optimal solutions and have a large amount of implicit parallelism. On the other hand, data clustering may be well formulated as a difficult global optimization problem; thereby making the application of SI tools more obvious and appropriate.

4 Clustering with the SI Algorithms

In this section we first review the present state of the art clustering algorithms based on SI tools, especially the ACO and PSO. We then outline a new algorithm which employs the PSO model to automatically determine the number of clusters in a previously unhandled dataset. Computer simulations undertaken for this study have also been included to demonstrate the elegance of the new dynamic clustering technique.

4.1 The Ant Colony Based Clustering Algorithms

Ant colonies provide a means to formulate some powerful nature-inspired heuristics for solving the clustering problems. Among other social movements, researchers have simulated the way, ants work collaboratively in the task of grouping dead bodies so, as to keep the nest clean (Bonabeau *et al.*, 1999). It can be observed that, with time the ants tend to cluster all dead bodies in a specific region of the environment, thus forming piles of corpses.

Larval sorting and corpse cleaning by ant was first modeled by Deneubourg *et al.* for accomplishing certain tasks in robotics (Deneubourg *et al.*, 1991). This inspired the Ant-based clustering algorithm (Handl *et al.*, 2003). Lumer and Faieta modified the algorithm using a dissimilarity-based evaluation of the local density, in order to make it suitable for data clustering (Lumer and Faieta, 1994). This introduced standard Ant Clustering Algorithm (ACA). It has subsequently been used for numerical data analysis (Lumer and Faieta,

1994), data-mining (Lumer and Faieta, 1995), graph-partitioning (Kuntz and Snyers, 1994, Kuntz and Snyers, 1999, Kuntz *et al.*, 1998) and text-mining (Handl and Meyer, 2002, Hoe *et al.*, 2002, Ramos and Merelo, 2002). Many authors (Handl and Meyer, 2002, Ramos *et al.*, 2002) proposed a number of modifications to improve the convergence rate and to get optimal number of clusters. Monmarche *et al.* hybridized the Ant-based clustering algorithm with K-means algorithm (Monmarche *et al.*, 1999) and compared it to traditional K-means on various data sets, using the classification error for evaluation purposes. However, the results obtained with this method are not applicable to ordinary ant-based clustering since it differs significantly from the latter.

Like a standard ACO, ant-based clustering is a distributed process that employs positive feedback. Ants are modeled by simple agents that randomly move in their environment. The environment is considered to be a low dimensional space, more generally a two-dimensional plane with square grid. Initially, each data object that represents a multi-dimensional pattern is randomly distributed over the 2-D space. Data items that are scattered within this environment can be picked up, transported and dropped by the agents in a probabilistic way. The picking and dropping operation are influenced by the similarity and density of the data items within the ant's local neighborhood. Generally, the size of the neighborhood is 3×3. Probability of picking up data items is more when the object are either isolated or surrounded by dissimilar items. They trend to drop them in the vicinity of similar ones. In this way, a clustering of the elements on the grid is obtained.

The ants search for the feature space either through random walk or with jumping using a short term memory. Each ant picks up or drops objects according to the following local probability density measure:

$$f(\mathbf{X}_i) = \max\{0, \frac{1}{s^2} \sum\nolimits_{\mathbf{X}_j \in N_{s\times s}(r)} [1 - \frac{d(\mathbf{X}_i, \mathbf{X}_j)}{\alpha(1 + \frac{\nu - 1}{\nu_{\max}})} \tag{14}$$

In the above expression, $N_{s\times s}(r)$ denotes the local area of perception surrounding the site of radius r, which the ant occupies in the two-dimensional grid. The threshold αg cales the dissimilarity within each pair of objects, and the moving speed v controls the step-size of the ant searching in the space within one time unit. If an ant is not carrying an object and finds an object $\mathbf{X}_i$ in its neighborhood, it picks up this object with a probability that is inversely proportional to the number of similar objects in the neighborhood. It may be expressed as:

$$P_{pick-up}(\mathbf{X}_i) = [\frac{k_p}{k_p + f(\mathbf{X}_i)}]^2 \tag{15}$$

If however, the ant is carrying an object x and perceives a neighbor's cell in which there are other objects, then the ant drops off the object it is carrying with a probability that is directly proportional to the object's similarity with the perceived ones. This is given by:

$$P_{drop}(\mathbf{X}_i) = \begin{matrix} 2.f(\mathbf{X}_i) & iff(\mathbf{X}_i) < k_d \\ 1 & iff(\mathbf{X}_i) \geq k_d \end{matrix}$$

The parameters k_p and k_d are the picking and dropping constants (Gath and Geva, 1989) respectively. Function $f(\mathbf{X}_i)$ provides an estimate of the density and similarity of elements in the neighborhood of object $\mathbf{X}_i$. The standard ACA pseudo-code is summarized in Algorithm 3.

Algorithm 3: Procedure ACA

```
Place every item X_i on a random cell of the grid;
Place every ant k on a random cell of the grid unoccupied by ants;
iteration_count ← 1;
while iteration_count < maximum_iteration do
  for i = 1 to no_of_ants do
    if unladen ant and cell occupied by item X_i then
      compute f(X_i) and P_pick-up(X_i);
    else
      if ant carrying item xi and cell empty then
        compute f(X_i) and P_drop(X_i);
        drop item X_i with probability P_drop(X_i);
      end if
    end if
    move to a randomly selected, neighboring and unoccupied cell ;
  end for
  t ← t + 1
end while
print location of items;
```

Kanade and Hall (Kanade and Hall, 2003) presented a hybridization of the ant systems with the classical FCM algorithm to determine the number of clusters in a given dataset automatically. In their fuzzy ant algorithm, at first the ant based clustering is used to create raw clusters and then these clusters are refined using the FCM algorithm. Initially the ants move the individual data objects to form heaps. The centroids of these heaps are taken as the initial cluster centers and the FCM algorithm is used to refine these clusters. In the second stage the objects obtained from the FCM algorithm are hardened according to the maximum membership criteria to form new heaps. These new heaps are then sometimes moved and merged by the ants. The final clusters formed are refined by using the FCM algorithm.

A number of modifications have been introduced to the basic ant based clustering scheme that improve the quality of the clustering, the speed of convergence and, in particular, the spatial separation between clusters on the grid, which is essential for the scheme of cluster retrieval. A detailed

description of the variants and results on the qualitative performance gains afforded by these extensions are provided in (Tsang and Kwong, 2006).

4.2 The PSO Based Clustering Algorithms

Research efforts have made it possible to view data clustering as an optimization problem. This view offers us a chance to apply PSO algorithm for evolving a set of candidate cluster centroids and thus determining a near optimal partitioning of the dataset at hand. An important advantage of the PSO is its ability to cope with local optima by maintaining, recombining and comparing several candidate solutions simultaneously. In contrast, local search heuristics, such as the simulated annealing algorithm (Selim and Alsultan, 1991) only refine a single candidate solution and are notoriously weak in coping with local optima. Deterministic local search, which is used in algorithms like the K-means, always converges to the nearest local optimum from the starting position of the search.

PSO-based clustering algorithm was first introduced by Omran *et al.* in (Omran *et al.*, 2002). The results of Omran *et al.* (Omran *et al.*, 2002,Omran *et al.*, 2005a) showed that PSO based method outperformed K-means, FCM and a few other state-of-the-art clustering algorithms. In their method, Omran *et al.* used a quantization error based fitness measure for judging the performance of a clustering algorithm. The quantization error is defined as:

$$J_e = \frac{\sum_{i=1}^{K} \sum_{\forall \mathbf{X}_j \in C_i} d(\mathbf{X}_j, \mathbf{V}_i)/n_i}{K} \tag{16}$$

where C_i is the i-th cluster center and n_i is the number of data points belonging to the i-th cluster. Each particle in the PSO algorithm represents a possible set of K cluster centroids as:

$\bar{Z}_i(t) =$

$\vec{V}_{i,1}$	$\vec{V}_{i,2}$	.....	$\vec{V}_{i,K}$

where $\mathbf{V}_{i,p}$ refers to the p-th cluster centroid vector of the i-th particle. The quality of each particle is measured by the following fitness function:

$$f(\mathbf{Z}_i, M_i) = w_1 \bar{d}_{\max}(M_i, \mathbf{X}_i) + w_2(R_{\max} - d_{\min}(\mathbf{Z}_i)) + w_3 J_e \tag{17}$$

In the above expression, R_{max} is the maximum feature value in the dataset and $\mathbf{M}_i$ is the matrix representing the assignment of the patterns to the clusters of the i-th particle. Each element $\mathrm{m}_{i,k,p}$indicates whether the pattern $\mathbf{X}_p$belongs to cluster C_k of i-th particle. The user-defined constants w_1, w_2,

and w_3 are used to weigh the contributions from different sub-objectives. In addition,

$$\bar{d}_{\max} = \max_{k \in 1,2,\ldots,K} \{ \sum_{\forall \mathbf{X}_p \in C_{i,K}} d(\mathbf{X}_p, \mathbf{V}_{i,k}) / n_{i,k} \} \tag{18}$$

and,

$$d_{\min}(\mathbf{Z}_i) = \min_{\forall p,q,p \neq q} \{ d(\mathbf{V}_{i,p}, \mathbf{V}_{i,q}) \} \tag{19}$$

is the minimum Euclidean distance between any pair of clusters. In the above, $n_{i,k}$is the number of patterns that belong to cluster $\mathbf{C}i,k$ of particle i. he fitness function is a multi-objective optimization problem, which minimizes the intra-cluster distance, maximizes inter-cluster separation, and reduces the quantization error. The PSO clustering algorithm is summarized in Algorithm 4.

Algorithm 4: The PSO Clustering Algorithm

```
Initialize each particle with K random cluster centers.
for iteration_count = 1 to maximum_iterations do
    for all particle i do
        for all pattern X_p in the dataset do
            calculate Euclidean distance of X_p with all cluster centroids
            assign X_p to the cluster that have nearest centroid to X_p
        end for
        calculate the fitness function f(Z_i, M_i)
    end for
    find the personal best and global best position of each particle.
    Update the cluster centroids according to velocity updating and coordinate
    updating formula of PSO.
end for
```

Van der Merwe and Engelbrecht hybridized this approach with the k-means algorithm for clustering general dataets (van der Merwe and Engelbrecht, 2003). A single particle of the swarm is initialized with the result of the k-means algorithm. The rest of the swarm is initialized randomly. In 2003, Xiao *et al* used a new approach based on the synergism of the PSO and the Self Organizing Maps (SOM) (Xiao *et al.*, 2003) for clustering gene expression data. They got promising results by applying the hybrid SOM-PSO algorithm over the gene expression data of Yeast and Rat Hepatocytes. Paterlini and Krink (Paterlini and Krink, 2006) have compared the performance of K-means, GA (Holland, 1975, Goldberg, 1975), PSO and Differential Evolution (DE) (Storn and Price, 1997) for a representative point evaluation approach to partitional clustering. The results show that PSO and DE outperformed the K-means algorithm.

Cui et al. (Cui and Potok, 2005) proposed a PSO based hybrid algorithm for classifying the text documents. They applied the PSO, K-means and a hybrid PSO clustering algorithm on four different text document datasets. The results illustrate that the hybrid PSO algorithm can generate more compact clustering results over a short span of time than the K-means algorithm.

4.3 An Automatic Clustering Algorithm Based on PSO

Tremendous research effort has gone in the past few years to evolve the clusters in complex datasets through evolutionary computing techniques. However, little work has been taken up to determine the optimal number of clusters at the same time. Most of the existing clustering techniques, based on evolutionary algorithms, accept the number of classes K as an input instead of determining the same on the run. Nevertheless, in many practical situations, the appropriate number of groups in a new dataset may be unknown or impossible to determine even approximately. For example, while clustering a set of documents arising from the query to a search engine, the number of classes K changes for each set of documents that result from an interaction with the search engine. Also if the dataset is described by high-dimensional feature vectors (which is very often the case), it may be practically impossible to visualize the data for tracking its number of clusters.

Finding an optimal number of clusters in a large dataset is usually a challenging task. The problem has been investigated by several researches (Halkidi *et al.*, 2001, Theodoridis and Koutroubas, 1999) but the outcome is still unsatisfactory (Rosenberger and Chehdi, 2000). Lee and Antonsson (Lee and Antonsson, 2000) used an Evolutionary Strategy (ES) (Schwefel, 1995) based method to dynamically cluster a dataset. The proposed ES implemented variable-length individuals to search for both centroids and optimal number of clusters. An approach to classify a dataset dynamically using Evolutionary Programming (EP) (Fogel *et al.*, 1966) can be found in Sarkar (Sarkar *et al.*, 1997) where two fitness functions are optimized simultaneously: one gives the optimal number of clusters, whereas the other leads to a proper identification of each cluster's centroid. Bandopadhyay *et al.* (Bandyopadhyay and Maulik, 2000) devised a variable string-length genetic algorithm (VGA) to tackle the dynamic clustering problem using a single fitness function. Very recently, Omran *et al.* came up with an automatic hard clustering scheme (Omran *et al.*, 2005c). The algorithm starts by partitioning the dataset into a relatively large number of clusters to reduce the effect of the initialization. Using binary PSO (Kennedy and Eberhart, 1997), an optimal number of clusters is selected. Finally, the centroids of the chosen clusters are refined through the K-means algorithm. The authors applied the algorithm for segmentation of natural, synthetic and multi-spectral images.

In this section we discuss a new fuzzy clustering algorithm (Das *et al.*, 2006), which can automatically determine the number of clusters in a given

dataset. The algorithm is based on a modified PSO algorithm with improved convergence properties.

The Modification of the Classical PSO

The canonical PSO has been subjected to empirical and theoretical investigations by several researchers (Eberhart and Shi, 2001, Clerc and Kennedy, 2002). In many occasions, the convergence is premature, especially if the swarm uses a small inertia weight ω or constriction coefficient (Clerc and Kennedy, 2002). As the global best found early in the searching process may be a poor local minima, we propose a multi-elitist strategy for searching the global best of the PSO. We call the new variant of PSO the MEPSO. The idea draws inspiration from the works reported in (Deb *et al.*, 2002). We define a growth rate β for each particle. When the fitness value of a particle of t-th iteration is higher than that of a particle of (t-1)-th iteration, the β will be increased. After the local best of all particles are decided in each generation, we move the local best, which has higher fitness value than the global best into the candidate area. Then the global best will be replaced by the local best with the highest growth rate β. Therefore, the fitness value of the new global best is always higher than the old global best. The pseudo code about MEPSO is described in Algorithm 5.

Algorithm 5: The MEPSO Algorithm

for $t = 1$ to t_{max} **do**
  if $t < t_{max}$ **then**
    for $j = 1$ to N **do** {swarm size is N}
      if the fitness value of $particle_j$ in t-th time-step > that of $particle_j$ in $(t-1)$-th time-step **then**
        $\beta_j = \beta_j + 1$;
      end if
      Update Local $best_j$.
      if the fitness of Local $best_j$ > that of Global best now **then**
        Choose Local $best_j$ put into candidate area.
      end if
    end for
    Calculate β of every candidate, and record the candidate of β_{max} .
    Update the Global best to become the candidate of β_{max}.
  else
    Update the Global best to become the particle of highest fitness value.
  end if
end for

Particle Representation

In the proposed method, for n data points, each p-dimensional, and for a user-specified maximum number of clusters c_{max}, a particle is a vector of real numbers of dimension $c_{max} + c_{max} \times p$. The first c_{max} entries are positive floating-point numbers in (0, 1), each of which controls whether the corresponding cluster is to be activated (i.e. to be really used for classifying the data) or not. The remaining entries are reserved for c_{max} cluster centers, each p-dimensional. A single particle can be shown as:

$\vec{Z}_i(t)$ =

$T_{i,1}$	$T_{i,2}$	.....	$T_{i,cmax}$	$\vec{V}_{i,1}$	$\vec{V}_{i,2}$	.....	$\vec{V}_{i,c_{max}}$
				$flag_{i,1}$	$flag_{i,2}$	.	$flag_{i,kmax}$

Activation Threshold **Cluster Centroids**

Every probable cluster center $m_{i,j}$ has p features and a binary $flag_{i,j}$ associated with it. The cluster center is active (i.e., selected for classification) if $flag_{i,j} = 1$ and inactive if $flag_{i,j} = 0$. Each flag is set or reset according to the value of the activation threshold $T_{i,j}$. Note that these flags are latent information associated with the cluster centers and do not take part in the PSO-type mutation of the particle. The rule for selecting the clusters specified by one particle is:

$$IfT_{i,j} > 0.5 Then flag_i, j = 1 Else flag_{i,j} = 0 \tag{20}$$

Note that the flags in an offspring are to be changed only through the T_{ij}'s (according to the above rule). When a particle jumps to a new position, according to (8), the T values are first obtained which then are used to select (via equation (6)) the m values. If due to mutation some threshold T in a particle exceeds 1 or becomes negative, it is fixed to 1 or zero, respectively. However, if it is found that no flag could be set to one in a particle (all activation thresholds are smaller than 0.5), we randomly select 2 thresholds and re-initialize them to a random value between 0.5 and 1.0. Thus the minimum number of possible clusters is 2.

Fitness Function

The quality of a partition can be judged by an appropriate cluster validity index. Cluster validity indices correspond to the statistical-mathematical functions used to evaluate the results of a clustering algorithm on a quantitative basis. Generally, a cluster validity index serves two purposes. First, it can

be used to determine the number of clusters, and secondly, it finds out the corresponding best partition. One traditional approach for determining the optimum number of classes is to run the algorithm repeatedly with different number of classes as input and then to select the partitioning of the data resulting in the best validity measure (Halkidi and Vazirgiannis, 2001). Ideally, a validity index should take care of the following aspects of the partitioning:

1. ***Cohesion***: Patterns in one cluster should be as similar to each other as possible. The fitness variance of the patterns in a cluster is an indication of the cluster's cohesion or compactness.
2. ***Separation*:** Clusters should be well separated. The distance among the cluster centers (may be their Euclidean distance) gives an indication of cluster separation.

In the present work we have based our fitness function on the Xie-Benni index. This index, due to (Xie and Beni, 1991), is given by:

$$XB_m = \frac{\sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^2 \left\| \mathbf{X}_j - \mathbf{V}_i \right\|^2}{n \times \min_{i \neq j} \left\| \mathbf{V}_i - \mathbf{V}_j \right\|^2} \tag{21}$$

Using XB_m the optimal number of clusters can be obtained by minimizing the index value. The fitness function may thus be written as:

$$f = \frac{1}{XB_i(c) + eps} \tag{22}$$

where XB_i is the Xie-Benni index of the i-th particle and eps is a very small constant (we used 0.0002). So maximization of this function means minimization of the XB index.

We have employed another famous validity index known as the partition entropy in order to judge the accuracy of the final clustering results obtained by MEPSO and its competitor algorithms in case of the image pixel classification. The partition entropy (Bezdek, 1981) function is given by,

$$V_{pe} = \frac{-\sum_{j=1}^{n} \sum_{i=1}^{c} [u_{ij} \log u_{ij}]}{n} \tag{23}$$

The idea of the validity function is that the partition with less fuzziness means better performance. Consequently, the best clustering is achieved when the value V_{pe} is minimal.

4.4 Avoiding Erroneous particles with Empty Clusters or Unreasonable Fitness Evaluation

There is a possibility that in our scheme, during computation of the XB index, a division by zero may be encountered. This may occur when one of

the selected cluster centers is outside the boundary of distributions of the data set. To avoid this problem we first check to see if any cluster has fewer than 2 data points in it. If so, the cluster center positions of this special chromosome are re-initialized by an average computation. We put n/c data points for every individual cluster center, such that a data point goes with a center that is nearest to it.

4.5 Combining All Together

The clustering method described here, is a two-pass process at each iteration or time step. The first pass amounts to calculating the active clusters as well as the membership functions for each particle in the spectral domain. In the second pass, the membership information of each pixel is mapped to the spatial domain, and the spatial function is computed from that. The MEPSO iteration proceeds with the new membership that is incorporated with the spatial function. The algorithm is stopped when the maximum number of time-steps t_{max} is exceeded. After the convergence, de-fuzzification is applied to assign each data item to a specific cluster for which the membership is maximal.

4.6 A Few Simulation Results

The MEPSO-clustering algorithm has been tested over a number of synthetic and real world datasets as well as on some image pixel classification problems. The performance of the method has been compared with the classical FCM algorithm and a recently developed fuzzy clustering algorithm based on GA. The later algorithm is referred in literature as Fuzzy clustering with Variable length Genetic Algorithm (FVGA) the details of which can be found in (Pakhira *et al.*, 2005). In the present chapter, we first provide the simulation results obtained over four well-chosen synthetic datasets (Bandyopadhyay and Maulik, 2000) and two real world datasets. The real world datasets used are the glass and the Wisconsin breast cancer data set, both of which have been taken from the UCI public data repository (Blake *et al.*, 1998). The glass data were sampled from six different type of glass: building windows float processed (70 objects), building windows non float processed (76 objects), vehicle windows float processed (17 objects), containers (13 objects), tableware (9 objects), headlamps (29 objects) with nine features each. The Wisconsin breast cancer database contains 9 relevant features: clump thickness, cell size uniformity, cell shape uniformity, marginal adhesion, single epithelial cell size, bare nuclei, bland chromatin, normal nucleoli and mitoses. The dataset has two classes. The objective is to classify each data vector into benign (239 objects) or malignant tumors (444 objects).

Performance of the MEPSO based algorithm on four synthetic datasets has been shown in Figures 5 through 8. In Table 1, we provide the mean value and standard deviation of the Xie Beni index evaluated over final clustering

results, the number of classes evaluated and the number of misclassified items with respect to the nominal partitions of the benchmark data, as known to us. For each data set, each run continues until the number of function evaluations (FEs) reaches 50,000. Twenty independent runs (with different seeds for the random number generator) have been taken for each algorithm. The results have been stated in terms of the mean *best-of-run* values and standard deviations over these 20 runs in each case. Only for the FCM, correct number of classes has been provided as input. Both FVGA and MEPSO determine the number of classes automatically on the run.

From Tables 1 and 2, one may see that our approach outperforms the state-of-the-art FVGA and the classical FCM over a variety of datasets in a statistically significant manner. Not only does the method find the optimal number of clusters, it also manages to find better clustering of the data points in terms of the two major cluster validity indices used in the literature.

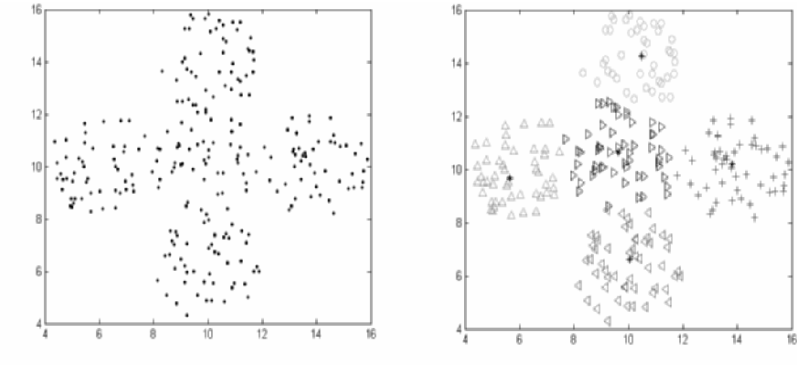

Fig. 5. (a) The unlabeled synthetic dataset 1 (b) Automatic Clustering with the MEPSO

4.7 Image Segmentation through Clustering

Image segmentation may be defined as the process of dividing an image into disjoint homogeneous regions. These homogeneous regions usually contain similar objects of interest or part of them. The extent of homogeneity of the segmented regions can be measured using some image property (e.g. pixel intensity (Jain *et al.*, 1999)). Segmentation forms a fundamental step towards several complex computer-vision and image analysis applications including digital mammography, remote sensing and land cover study. Image segmentation can be treated as a clustering problem where the features describing each pixel correspond to a pattern, and each image region (i.e., segment)

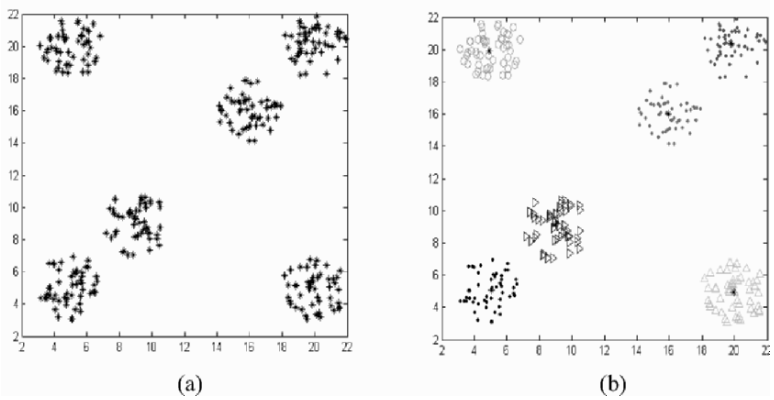

Fig. 6. (a) The unlabeled synthetic dataset 1 (b) Automatic Clustering with the MEPSO

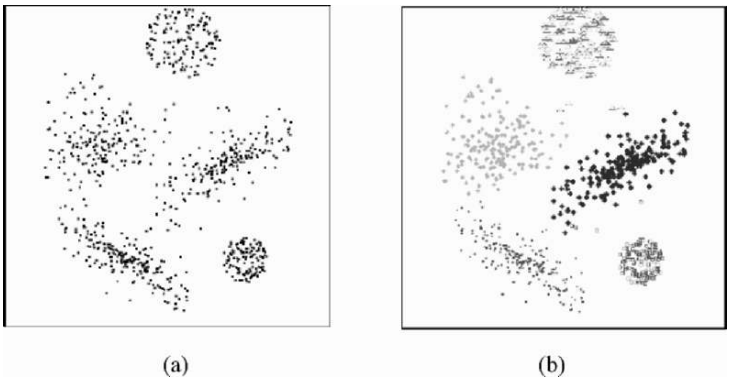

Fig. 7. (a) The unlabeled synthetic dataset 1 (b) Automatic Clustering with the MEPSO

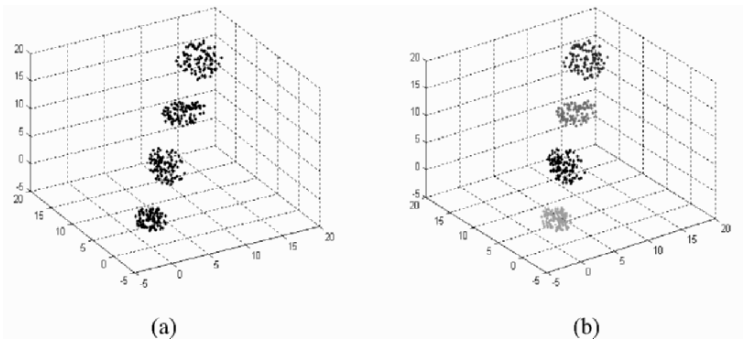

Fig. 8. (a) The unlabeled synthetic dataset 1 (b) Automatic Clustering with the MEPSO

Table 1. Final solution (mean and standard deviation over 20 independent runs) after each algorithm was terminated after running for 50,000 function evaluations (FE) with DB Measure based fitness function.

Problem	Algorithm	Average no. of clusters found	Final DB measure	Mean No. of misclassified Items
Synthetic Data1	MEPSO	**5.05±0.0931**	**3.0432±0.021**	**5.25±0.096**
	FVGA	8.15±0.0024	4.3432±0.232	15.75±0.154
	FCM	NA	5.3424±0.343	19.50±1.342
Synthetic Data2	MEPSO	**6.45±0.0563**	**1.4082±0.006**	**4.50±0.023**
	FVGA	6.95±0.021	1.5754±0.073	10.25±0.373
	FCM	NA	1.6328±0.002	26.50±0.433
Synthetic Data3	MEPSO	**5.25±0.0241**	**0.9224±0.334**	**9.15±0.034**
	FVGA	5.75±0.0562	1.2821±0.009	15.50±0.048
	FCM	NA	2.9482±0.028	17.25±0.275
Synthetic Data4	MEPSO	**4.00±0.00**	**1.0092±0.083**	**1.50±0.035**
	FVGA	4.75±0.0193	1.5152±0.073	4.55±0.05
	FCM	NA	1.8371±0.034	8.95±0.15
Glass	MEPSO	**6.05±0.0248**	**1.0802±0.083**	**8.35±0.662**
	FVGA	5.95±0.0193	1.5152±0.073	14.35±0.26
	FCM	NA	1.8371±0.034	18.65±0.85
Breast Cancer	MEPSO	2.05±0.0563	0.5003±0.006	**25.00±0.09**
	FVGA	2.50±0.0621	0.5754±0.073	26.50±0.80
	FCM	NA	0.6328±0.002	30.23±0.46

corresponds to a cluster (Jain *et al.*, 1999). Therefore, many clustering algorithms have widely been used to solve the segmentation problem (e.g., K-means (Tou and Gonzalez, 1974), Fuzzy C-means (Trivedi and Bezdek, 1986), ISODATA (Ball and Hall, 1967), Snob (Wallace and Boulton, 1968) and recently the PSO and DE based clustering techniques (Omran *et al.*, 2005a, Omran *et al.*, 2005b)).

Here we illustrate the automatic soft segmentation of a number of grey scale images by using our MEPSO based clustering algorithm. An important characteristic of an image is the high degree of correlation among the neighboring pixels. In other words, these neighboring pixels possess similar feature values, and the probability that they belong to the same cluster is great. This spatial relationship (Ahmed *et al.*, 2002) is important in clustering, but it is not utilized in a standard FCM algorithm. To exploit the spatial information, a spatial function is defined as:

$$h_{ij} = \sum_{k \in \delta(\mathbf{X}_j)} u_{ik} \tag{24}$$

where $\delta(\mathbf{X}_j)$represents a square window centered on pixel (i.e. data point) $\mathbf{X}_j$ in the spatial domain. A 5×5 window was used throughout this work. Just like the membership function, the spatial function h_{ij} represents the probability

that pixel $\mathbf{X}_j$ belongs to i-th cluster. The spatial function of a pixel for a cluster is large if the majority of its neighborhood belongs to the same clusters. We incorporate the spatial function into membership function as follows:

$$u'_{ij} = \frac{u_{ij}^r h_{ij}^t}{\sum_{k=1}^{c} u_{kj}^r h_{kj}^t} \tag{25}$$

Here in all the cases we have used r = 1, t = 1after considerable trial and errors.

Although we tested our algorithm over a large number of images with varying range of complexity, here we show the experimental results for three images only, due to economy of space. Figures 4.7 to 4.7 show the three original images and their segmented counterparts obtained using the FVGA algorithm and the MEPSO based method. In these figures the segmented portions of an image have been marked with the grey level intensity of the respective cluster centers. In Table 2, we report the mean value the DB measure and partition entropy calculated over the 'best-of-run' solutions in each case. One may note that the MEPSO meets or beats the competitor algorithm in all the cases. Table 3 reports the mean time taken by each algorithm to terminate on the image data. Finally, Table 4 contains the mean and standard deviations of the number of classes obtained by the two automatic clustering algorithms.

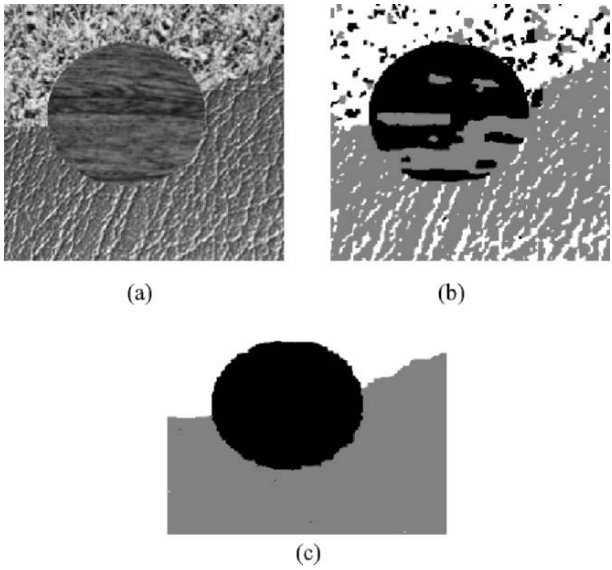

Fig. 9. (a) The original Texture image. (b) Segmentation by FVGA (c= 3) (c) Segmentation by MEPSO based method (c = 3)

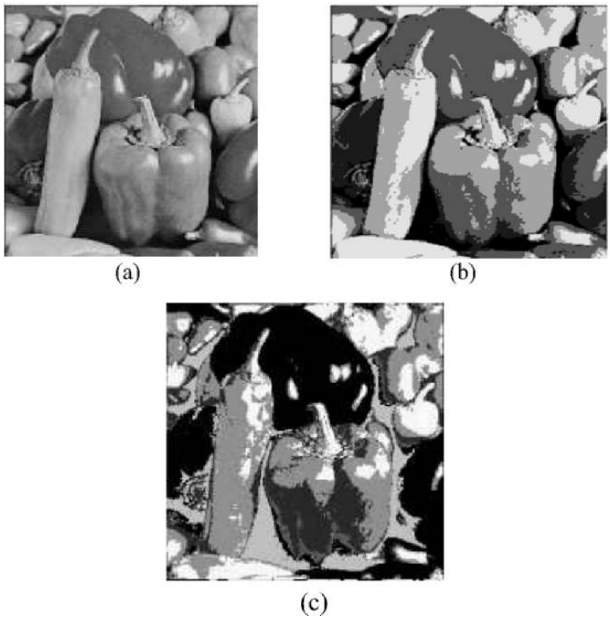

Fig. 10. (a) The original Pepper image. (b) Segmentation by FVGA (c= 7) (c) Segmentation by MEPSO based method (c = 7)

Table 2. Automatic clustering result for three real life grayscale images (over 20 runs; each run continued up to 50,000 FE)

Image	Validity Index	Mean and Std Dev of the validity indices over the final clustering results of 20 independent runs		
		AFDE	FVGA	FCM
Texture	Xie-Beni	**0.7283 (0.0001)**	0.7902 (0.0948)	0.7937 (0.0013)
	Partition Entropy	**2.6631 (0.7018)**	2.1193 (0.8826)	2.1085 (0.0043)
MRI Image of Brain	Xie-Beni	**0.2261 (0.0017)**	0.2919 (0.0583)	0.3002 (0.0452)
	Partition Entropy	**0.1837 (0.0017)**	0.1922 (0.0096)	0.1939 (0.0921)
Pepper Image	Xie-Beni	**0.05612 (0.0092)**	0.09673 (0.0043)	0.09819 (0.0001)
	Partition Entropy	**0.8872 (0.0137)**	1.1391 (0.0292)	1.1398 (0.0884)

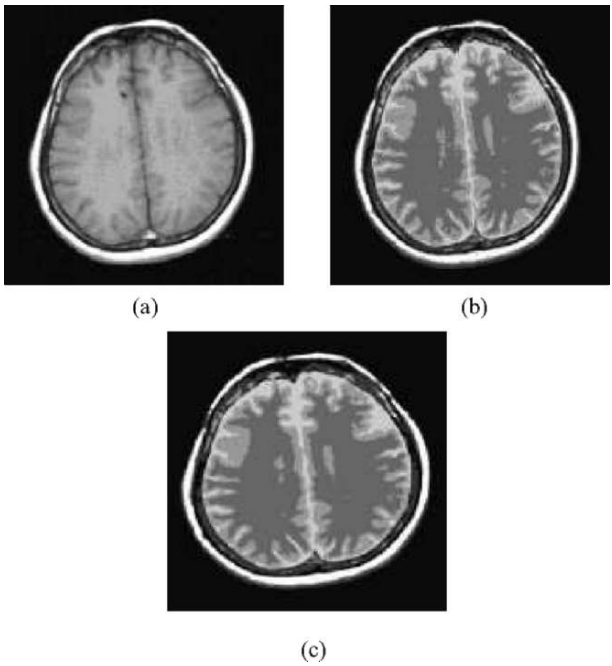

Fig. 11. (a) The original MRI image. (b) Segmentation by FVGA (c= 5) (c) Segmentation by MEPSO (c = 5)

Table 3. Comparison among the mean execution time taken by the different algorithms

Image	Optimal No. of Clusters	Mean and Std Dev of the number of classes estimated by the competitor algorithms	
		FVGA	MEPSO
Texture	3	3.75±0.211	3.05±0.132
MRI	5	5.05±0.428	5.25±0.212
Pepper	7	8.15±0.772	6.95±0.982

Table 4. Automatic clustering results for the three real-life grayscale images (over 20 runs; each runs continued for 50,000 FE)

Image	Mean and Std Dev of the execution time (in seconds) taken by the competitor algorithms	
	FVGA	MEPSO
Texture	32.05±0.076	47.25±0.162
MRI	24.15±0.016	34.65±0.029
Pepper	49.20±0.201	67.85±0.817

5 Conclusion and Future Directions

In this Chapter, we introduced some of the preliminary concepts of Swarm Intelligence (SI) with an emphasis on particle swarm optimization and ant colony optimization algorithms. We then described the basic data clustering terminologies and also illustrated some of the past and ongoing works, which apply different SI tools to pattern clustering problems. We proposed a novel fuzzy clustering algorithm, which is based on a deviant variety of the PSO. The proposed algorithm can automatically compute the optimal number of clusters in any dataset and thus requires minimal user intervention. Comparison with a state of the art GA based clustering strategy, reveals the superiority of the MEPSO-clustering algorithm both in terms of accuracy and speed.

Despite being an age old problem, clustering remains an active field of interdisciplinary research till date. No single algorithm is known, which can group all real world datasets efficiently and without error. To judge the quality of a clustering, we need some specially designed statistical-mathematical function called the clustering validity index. But a literature survey reveals that, most of these validity indices are designed empirically and there is no universally good index that can work equally well over any dataset. Since, majority of the PSO or ACO based clustering schemes rely on a validity index to judge the fitness of several possible partitioning of the data, research effort should be spent for defining a reasonably good index function and validating the same mathematically.

Feature extraction is an important preprocessing step for data clustering. Often we have a great number of features (especially for a high dimensional dataset like a collection of text documents) which are not all relevant for a given operation. Hence, future research may focus on integrating the automatic feature-subset selection scheme with the SI based clustering algorithm. The two-step process is expected to automatically project the data to a low dimensional feature subspace, determine the number of clusters and find out the appropriate cluster centers with the most relevant features at a faster pace.

Gene expression refers to a process through which the coded information of a gene is converted into structures operating in the cell. It provides the physical evidence that a gene has been "turned on" or activated for protein synthesis (Lewin, 1995). Proper selection, analysis and interpretation of the gene expression data can lead us to the answers of many important problems in experimental biology. Promising results have been reported in (Xiao *et al.*, 2003) regarding the application of PSO for clustering the expression levels of gene subsets. The research effort to integrate SI tools in the mechanism of gene expression clustering may in near future open up a new horizon in the field of bioinformatic data mining.

Hierarchical clustering plays an important role in fields like information retrieval and web mining. The self-assembly behavior of the real ants may be exploited to build up new hierarchical tree-structured partitioning of a

data set according to the similarities between those data items. A description of the little but promising work already been undertaken in this direction can be found in (Azzag *et al.*, 2006). But a more extensive and systematic research effort is necessary to make the ant based hierarchical models superior to existing algorithms like Birch (Zhang *et al.*, 1997).

References

A. Abraham, C. Grosan and V. Ramos (2006) (Eds.), Swarm Intelligence and Data Mining, Studies in Computational Intelligence, Springer Verlag, Germany, pages 270, ISBN: 3-540-34955-3.

Ahmed MN, Yaman SM, Mohamed N, (2002), Farag AA and Moriarty TA, Modified fuzzy c-means algorithm for bias field estimation and segmentation of MRI data. IEEE Trans Med Imaging, 21, pp. 193–199.

Azzag H, Guinot C and Venturini G, Data and text mining with hierarchical clustering ants, in *Swarm Intelligence in Data Mining*, Abraham A, (2006), Grosan C and Ramos V (Eds), Springer, pp. 153-186.

Ball G and Hall D, (1967), A Clustering Technique for Summarizing Multivariate Data, Behavioral Science 12, pp. 153-155.

Bandyopadhyay S and Maulik U, (2000), Genetic clustering for automatic evolution of clusters and application to image classification, Pattern Recognition, 35, pp. 1197-1208.

Beni G and Wang U, (1989), Swarm intelligence in cellular robotic systems. In *NATO Advanced Workshop on Robots and Biological Systems*, Il Ciocco, Tuscany, Italy.

Bensaid AM, Hall LO, Bezdek JC.and Clarke LP, (1996), Partially supervised clustering for image segmentation. Pattern Recognition, vol. 29, pp. 859-871.

Bezdek JC, (1981), Pattern recognition with fuzzy objective function algorithms. New York: Plenum.

Blake C, Keough E and Merz CJ, (1998), UCI repository of machine learning database http://www.ics.uci.edu/∼mlearn/MLrepository.html.

Bonabeau E, Dorigo M and Theraulaz G, (1999), *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, New York.

Brucker P, (1978), On the complexity of clustering problems. Beckmenn M and Kunzi HP(Eds.), *Optimization and Operations Research*, Lecture Notes in Economics and Mathematical Systems, Berlin, Springer, vol.157, pp. 45-54.

Clark MC, Hall LO, Goldgof DB, Clarke LP, (1994), Velthuizen RP and Silbiger MS , MRI segmentation using fuzzy clustering techniques. IEEE Eng Med Biol, 13, pp.730–742.

Clerc M and Kennedy J. (2002), The particle swarm - explosion, stability, and convergence in a multidimensional complex space, In IEEE Transactions on Evolutionary Computation, 6(1):58-73.

Couzin ID, Krause J, James R, Ruxton GD, Franks NR, (2002), Collective Memory and Spatial Sorting in Animal Groups, Journal of Theoretical Biology, 218, pp. 1-11.

Cui X and Potok TE, (2005), Document Clustering Analysis Based on Hybrid PSO+Kmeans Algorithm, Journal of Computer Sciences (Special Issue), ISSN 1549-3636, pp. 27-33.

Das S, Konar A and Abraham A, (2006), Spatial Information based Image Segmentation with a Modified Particle Swarm Optimization, in proceedings of Sixth International Conference on Intelligent System Design and Applications (ISDA 06) Jinan, Shangdong, China, IEEE Computer Society Press.

Deb K, Pratap A, Agarwal S, and Meyarivan T (2002), A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. on Evolutionary Computation, Vol.6, No.2.

Deneubourg JL, Goss S, Franks N, Sendova-Franks A, (1991), Detrain C and Chetien L , The dynamics of collective sorting: Robot-like ants and ant-like robots. In Meyer JA and Wilson SW (Eds.) *Proceedings of the First International Conference on Simulation of Adaptive Behaviour: From Animals to Animats 1*, pp. 356–363. MIT Press, Cambridge, MA.

Dorigo M and Gambardella LM, (1997), Ant colony system: A cooperative learning approach to the traveling salesman problem, IEEE Trans. Evolutionary Computing, vol. 1, pp. 53–66.

Dorigo M, Maniezzo V and Colorni A, (1996), The ant system: Optimization by a colony of cooperating agents, IEEE Trans. Systems Man and Cybernetics – Part B, vol. 26.

Duda RO and Hart PE, (1973), *Pattern Classification and Scene Analysis*. John Wiley and Sons, USA.

Eberhart RC and Shi Y, (2001), Particle swarm optimization: Developments, applications and resources, In Proceedings of IEEE International Conference on Evolutionary Computation, vol. 1, pp. 81-86.

Evangelou IE, Hadjimitsis DG, Lazakidou AA, (2001), Clayton C, Data Mining and Knowledge Discovery in Complex Image Data using Artificial Neural Networks, Workshop on Complex Reasoning an Geographical Data, Cyprus.

Everitt BS, (1993), *Cluster Analysis*. Halsted Press, Third Edition.

Falkenauer E, (1998), *Genetic Algorithms and Grouping Problems*, John Wiley and Son, Chichester.

Fogel LJ, Owens AJ and Walsh MJ, (1966), Artificial Intelligence through Simulated Evolution. New York: Wiley.

Forgy EW, (1965), Cluster Analysis of Multivariate Data: Efficiency versus Interpretability of classification, Biometrics, 21.

Frigui H and Krishnapuram R, (1999), A Robust Competitive Clustering Algorithm with Applications in Computer Vision, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (5), pp. 450-465.

Fukunaga K, (1990), Introduction to Statistical Pattern Recognition. Academic Press.

Gath I and Geva A, (1989), Unsupervised optimal fuzzy clustering. IEEE Transactions on PAMI, 11, pp. 773-781.

Goldberg DE, (1975), Genetic *Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA.

Grosan C, Abraham A and Monica C, Swarm Intelligence in Data Mining, in *Swarm Intelligence in Data Mining*, Abraham A, (2006), Grosan C and Ramos V (Eds), Springer, pp. 1-16.

Halkidi M and Vazirgiannis M, (2001), Clustering Validity Assessment: Finding the Optimal Partitioning of a Data Set. Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM 01), San Jose, California, USA, pp. 187-194.

Halkidi M, Batistakis Y and Vazirgiannis M, (2001), On Clustering Validation Techniques. Journal of Intelligent Information Systems (JIIS), 17(2-3), pp. 107-145.

Handl J and Meyer B, (2002), Improved ant-based clustering and sorting in a document retrieval interface. In Proceedings of the Seventh International Conference on Parallel Problem Solving from Nature (PPSN VII), volume 2439 of LNCS, pp. 913–923. Springer-Verlag, Berlin, Germany.

Handl J, Knowles J and Dorigo M, (2003), Ant-based clustering: a comparative study of its relative performance with respect to k-means, average link and 1D-som. Technical Report TR/IRIDIA/2003-24. IRIDIA, Universite Libre de Bruxelles, Belgium.

Hoe K, Lai W, and Tai T, (2002), Homogenous ants for web document similarity modeling and categorization. In *Proceedings of the Third International Workshop on Ant Algorithms (ANTS 2002)*, volume 2463 of *LNCS*, pp. 256–261. Springer-Verlag, Berlin, Germany.

Holland JH, (1975), *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor.

Jain AK, Murty MN and Flynn PJ, (1999), Data clustering: a review, ACM Computing Surveys, vol. 31, no.3, pp. 264—323.

Kanade PM and Hall LO, (2003), Fuzzy Ants as a Clustering Concept. In Proceedings of the 22nd International Conference of the North American Fuzzy Information Processing Society (NAFIPS03), pp. 227-232.

Kaufman, L and Rousseeuw, PJ, (1990), *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, New York.

Kennedy J and Eberhart R, (1995), Particle swarm optimization, In Proceedings of IEEE International conference on Neural Networks, pp. 1942-1948.

Kennedy J and Eberhart RC, (1997), A discrete binary version of the particle swarm algorithm, Proceedings of the 1997 Conf. on Systems, Man, and Cybernetics, IEEE Service Center, Piscataway, NJ, pp. 4104-4109.

Kennedy J, Eberhart R and Shi Y, (2001), *Swarm Intelligence*, Morgan Kaufmann Academic Press.

Kohonen T, (1995), *Self-Organizing Maps*, Springer Series in Information Sciences, Vol 30, Springer-Verlag.

Konar A, (2005), Computational Intelligence: Principles, Techniques and Applications, Springer.

Krause J and Ruxton GD, (2002), Living in Groups. Oxford: Oxford University Press.

Kuntz P and Snyers D, (1994), Emergent colonization and graph partitioning. In *Proceedings of the Third International Conference on Simulation of Adaptive Behaviour: From Animals to Animats 3*, pp. 494– 500. MIT Press, Cambridge, MA.

Kuntz P and Snyers D, (1999), New results on an ant-based heuristic for highlighting the organization of large graphs. In *Proceedings of the 1999 Congress on Evolutionary Computation*, pp. 1451–1458. IEEE Press, Piscataway, NJ.

Kuntz P, Snyers D and Layzell P, (1998), A stochastic heuristic for visualising graph clusters in a bi-dimensional space prior to partitioning. *Journal of Heuristics*, 5(3), pp. 327–351.

Lee C-Y and Antonsson EK, (2000), Self-adapting vertices for mask layout synthesis Modeling and Simulation of Microsystems Conference (San Diego, March

27–29) eds. M Laudon and B Romanowicz. pp. 83–86.
Leung Y, Zhang J and Xu Z, (2000), Clustering by Space-Space Filtering, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (12), pp. 1396-1410.
Lewin B, (1995), *Genes VII*. Oxford University Press, New York, NY.
Lillesand T and Keifer R, (1994), *Remote Sensing and Image Interpretation*, John Wiley & Sons, USA.
Lumer E and Faieta B, (1994), Diversity and Adaptation in Populations of Clustering Ants. In Proceedings Third International Conference on Simulation of Adaptive Behavior: from animals to animates 3, Cambridge, Massachusetts MIT press, pp. 499-508.
Lumer E and Faieta B, (1995), Exploratory database analysis via self-organization, Unpublished manuscript.
MacQueen J, (1967), Some methods for classification and analysis of multivariate observations, Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp. 281-297.
Major PF, Dill LM, (1978), The three-dimensional structure of airborne bird flocks. Behavioral Ecology and Sociobiology, 4, pp. 111-122.
Mao J and Jain AK, (1995), Artificial neural networks for feature extraction and multivariate data projection. IEEE Trans. Neural Networks.vol. 6, 296–317.
Milonas MM, (1994), Swarms, phase transitions, and collective intelligence, In Langton CG Ed., Artificial Life III, Addison Wesley, Reading, MA.
Mitchell T, (1997), *Machine Learning*. McGraw-Hill, Inc., New York, NY.
Mitra S, Pal SK and Mitra P, (2002), Data mining in soft computing framework: A survey, IEEE Transactions on Neural Networks, Vol. 13, pp. 3-14.
Monmarche N, Slimane M and Venturini G, (1999), Ant Class: discovery of clusters in numeric data by a hybridization of an ant colony with the k means algorithm. Internal Report No. 213, E3i, Laboratoire d'Informatique, Universite de Tours.
Ng R and Han J, (1994), Efficient and effective clustering method for spatial data mining. In: Proc. 1994 International Conf. Very Large Data Bases (VLDB'94). Santiago, Chile, September pp. 144–155.
Omran M, Engelbrecht AP and Salman A, (2005), Particle Swarm Optimization Method for Image Clustering. *International Journal of Pattern Recognition and Artificial Intelligence*, 19(3), pp. 297–322.
Omran M, Engelbrecht AP and Salman A, (2005), Differential Evolution Methods for Unsupervised Image Classification, Proceedings of Seventh Congress on Evolutionary Computation (CEC-2005). IEEE Press.
Omran M, Salman A and Engelbrecht AP, (2002), Image Classification using Particle Swarm Optimization. In *Conference on Simulated Evolution and Learning*, volume 1, pp. 370–374.
Omran M, Salman A and Engelbrecht AP, (2005), Dynamic Clustering using Particle Swarm Optimization with Application in Unsupervised Image Classification. Fifth World Enformatika Conference (ICCI 2005), Prague, Czech Republic.
Pakhira MK, Bandyopadhyay S and Maulik, U, (2005), A Study of Some Fuzzy Cluster Validity Indices, Genetic clustering And Application to Pixel Classification, Fuzzy Sets and Systems 155, pp. 191–214.
Pal NR, Bezdek JC and Tsao ECK, (1993), Generalized clustering networks and Kohonen's self-organizing scheme. IEEE Trans. Neural Networks, vol 4, 549–557.

Partridge BL, (1982), The structure and function of fish schools. Science American, 245, pp. 90-99.

Partridge BL, Pitcher TJ, (1980), The sensory basis of fish schools: relative role of lateral line and vision. Journal of Comparative Physiology, 135, pp. 315-325.

Paterlini S and Krink T, (2006), Differential Evolution and Particle Swarm Optimization in Partitional Clustering. *Computational Statistics and Data Analysis*, vol. 50, pp. 1220– 1247.

Paterlini S and Minerva T, (2003), Evolutionary Approaches for Cluster Analysis. In Bonarini A, Masulli F and Pasi G (eds.) *Soft Computing Applications*. Springer-Verlag, Berlin. 167-178.

Ramos V and Merelo JJ, (2002), Self-organized stigmergic document maps: Environments as a mechanism for context learning. In *Proceedings of the First Spanish Conference on Evolutionary and Bio-Inspired Algorithms (AEB 2002)*, pp. 284–293. Centro Univ. M'erida, M'erida, Spain.

Ramos V, Muge F and Pina P, (2002), Self-Organized Data and Image Retrieval as a Consequence of Inter-Dynamic Synergistic Relationships in Artificial Ant Colonies. Soft Computing Systems: Design, Management and Applications. 87, pp. 500–509.

Rao MR, (1971), Cluster Analysis and Mathematical Programming,. Journal of the American Statistical Association, Vol. 22, pp 622-626.

Rokach, L., Maimon, O. (2005), Clustering Methods, Data Mining and Knowledge Discovery Handbook, Springer, pp. 321-352.

Rosenberger C and Chehdi K, (2000), Unsupervised clustering method with optimal estimation of the number of clusters: Application to image segmentation, in Proc. IEEE International Conference on Pattern Recognition (ICPR), vol. 1, Barcelona, pp. 1656-1659.

Sarkar M, Yegnanarayana B and Khemani D, (1997), A clustering algorithm using an evolutionary programming-based approach, Pattern Recognition Letters, 18, pp. 975–986.

Schwefel H-P, (1995), *Evolution and Optimum Seeking*. New York, NY: Wiley, 1st edition.

Selim SZ and Alsultan K, (1991), A simulated annealing algorithm for the clustering problem. Pattern recognition, 24(7), pp. 1003-1008.

Storn R and Price K, (1997), Differential evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, Journal of Global Optimization, 11(4), pp. 341–359.

Theodoridis S and Koutroubas K, (1999), Pattern recognition, Academic Press.

Tou JT and Gonzalez RC, (1974), Pattern Recognition Principles. London, Addison-Wesley.

Trivedi MM and Bezdek JC, (1986), Low-level segmentation of aerial images with fuzzy clustering, IEEE Trans.on Systems, Man and Cybernetics, Volume 16.

Tsang W and Kwong S, Ant Colony Clustering and Feature Extraction for Anomaly Intrusion Detection, in *Swarm Intelligence in Data Mining*, Abraham A, (2006), Grosan C and Ramos V (Eds), Springer, pp. 101-121.

van der Merwe DW and Engelbrecht AP, (2003), Data clustering using particle swarm optimization. In: Proceedings of the 2003 IEEE Congress on Evolutionary Computation, pp. 215-220, Piscataway, NJ: IEEE Service Center.

Wallace CS and Boulton DM, (1968), An Information Measure for Classification, Computer Journal, Vol. 11, No. 2, 1968, pp. 185-194.

Wang X, Wang Y and Wang L, (2004), Improving fuzzy c-means clustering based on feature-weight learning. Pattern Recognition Letters, vol. 25, pp. 1123–32.

Xiao X, Dow ER, Eberhart RC, Miled ZB and Oppelt RJ, (2003), Gene Clustering Using Self-Organizing Maps and Particle Swarm Optimization, Proc of the 17th International Symposium on Parallel and Distributed Processing (PDPS '03), IEEE Computer Society, Washington DC.

Xie, X and Beni G, (1991), Validity measure for fuzzy clustering. IEEE Trans. Pattern Anal. Machine Learning, Vol. 3, pp. 841–846.

Xu, R., Wunsch, D. (2005), Survey of Clustering Algorithms, IEEE Transactions on Neural Networks, Vol. 16(3): 645-678.

Zahn CT, (1971), Graph-theoretical methods for detecting and describing gestalt clusters, IEEE Transactions on Computers C-20, 68–86.

Zhang T, Ramakrishnan R and Livny M, (1997), BIRCH: A New Data Clustering Algorithm and Its Applications, Data Mining and Knowledge Discovery, vol. 1, no. 2, pp. 141-182.

Hall LO, Özyurt IB and Bezdek JC, (1999), Clustering with a genetically optimized approach, IEEE Trans. Evolutionary Computing 3 (2) pp. 103–112.