NATURAL-INSPIRED DATA CLUSTERING: A HYBRIDIZATION BETWEEN ANT CLUSTERING AND PARTICLE SWARM OPTIMIZATION

Mohammed El-Telbany
Computer Engineering Department
Taif University, Saudi Arabia
telbany@eri.sci.eg

Samah Refat
Faculty of Arts and Science
Ain Shams University, Egypt
samah_refat2006@hotmail.com

Sultan H. Aljahdali
Computer Science Department
Taif University, Saudi Arabia
aljahdali@tu.edu.sa

Abstract

The clustering algorithms have evolved over the last decade. With the continuous success of natural inspired algorithms in solving many engineering problems, it is imperative to scrutinize the success of these methods applied to data clustering. These naturally inspired algorithms are mainly stochastic search and optimization techniques, guided by the principles of collective behavior and self-organization of insect swarms. The parameters setting of the ant colony clustering algorithms determine the behavior of each ant and are critical for fast convergence to near optimal solutions of clustering task. This inspired us to explore techniques for automatically learning the optimal parameters for a given clustering task. We devised and implemented a hybrid Ant-Colony clustering algorithm, which uses particle swarm optimization algorithm in the early stages to 'breed' a population of ants possessing near optimal behavioral parameter settings for a given problem. This hybrid algorithm converges rapidly for nearly optimal parameters that maximize the ant-colony clustering behavior.

1 INTRODUCTION

Natural inspired artificial intelligence algorithms—and machine learning in general—provide us with methods and techniques for discovering knowledge. These natural-inspired algorithms imitate nature in one way or another. These natural entities have competed for resources to ensure their survival or collectively cooperate to exhibit impressive problem solving skills. Neural networks imitate the structure of our human brain and genetic algorithms simulate evolution to name just two. They are characterized by inherent parallelism, adaptively, positive feedback and some element of randomness. The evolutionary algorithms simulate natural evolution that based on natural selection and genetics by combining the fittest individual such as genetic algorithms, evolutionary strategies. The swarm intelligence originally took its inspiration from the biological examples by swarming, flocking and herding phenomena in vertebrates.

These powerful algorithms can be used for prediction, classification, and clustering and have clear application for use in financial modeling, image processing and web-mining. Clustering is often used as a tool for preliminary and descriptive data analysis and for unsupervised classification. Its main purpose is to identify homogeneous groups by finding similarities between objects regarding their characterizing attributes. 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*. Clustering is widely applied methodology in many applications include statistics, mathematical programming (such as location selecting, network partitioning, routing, scheduling and assignment problems, etc.) and computer science (including pattern recognition, learning theory, image processing and computer graphics, etc.). Clustering is mainly to group all objects into several mutually exclusive clusters in order to optimize an objective function. Formally, the clustering problem can be formulated an *optimization* and *search* problem where the result is to find the partition $P^* = \{C_1, \ldots, C_k\}$ that has optimal adequacy with respect to all other feasible solutions $\{P^1, \ldots, P^{S(m,K)}\}$. Where $S(m,K)$ denote the number of possible clustering of m vectors into K groups. This is equivalent to a mathematical function $J_e(P)$ that needs to be optimized, where $J_e(P)$ represents the quality measurement for a partition p given $\forall P \; J_e(P) \geq 0$. The problem is to find the best solution (i.e., partition) P^* such that:

$$J_e(P^*) = \underset{P}{Max}\, J_e(P) \tag{1}$$

There are many different ways to express and formulate the clustering problem; as a consequence, the obtained results and its interpretations depend strongly on the way the clustering problem was originally formulated. For example, the clusters or groups that are identified may be exclusive, so that every instance belongs in only one group (i.e., *hard clustering*). Or, they may be overlapping, meaning that one instance may fall into several clusters (i.e., *soft clustering*). Or they may be *probabilistic*, whereby an instance belongs to each group depending on a certain assigned probability. Such statistical approaches make sense in practical situations where no amount of training data is sufficient to make a completely firm decision about cluster memberships. Or they may be *hierarchical*, such that there is a crude division of the instances into groups at a high level that is further refined into a finer levels. Furthermore, different formulations lead to different algorithms to solve. If we also consider all the "*variations*" of each different algorithm proposed to solve each different formulation, we end up with a very large family of clustering algorithms [18]. Each algorithm has its own approach for handling cluster validity, number of clusters, and structure imposed on the data. For many years now, several papers have highlighted the efficiency of approaches using nature inspired algorithms. In particular varieties of these algorithms have inspired by the evolutionary algorithms and swarm intelligence. The swarm intelligence techniques as promising techniques have been applied to clustering optimization problems, the particle swarm optimization (PSO) and ant colony optimization (ACO) [4, 5, 21, 22]. *Ant-inspired*, *distributed* or *agent-based* clustering is a technique inspired by corpse clustering and brood sorting behavior of certain ant species. Clustering can simply mean aggregating objects into piles, but usually the organization into groups of similar objects is meant [1, 2,10]. Unlike in classification tasks, the groups are not known a priori. Distributed clustering works with simulated ants *(agents)*, which move over a grid, pickup scattered data elements and drop them in areas with a high concentration of similar elements. However, the ant-clustering algorithms are characterized by a number of controlling parameters such as the number of iterations of the algorithm; the number of agents (ants) used per iteration, and while the meaning of each parameter is known, their sensitivities or how they interact with each other is not. Small changes in one variable compose noticeable, but unpredictable changes in results, so that hand-tuning an ant-clustering algorithm is a daunting task. To solve this problem, the use of particle swarm optimization algorithm is proposed to automatically choose the best parameters for a given task. The rest of the paper is organized in the following manner. In section 2, a brief review of the works carried out in the area of hybridization the ant-clustering algorithms in research are presented. In section 3 and 4, the ant-based clustering and particle swarm optimization algorithms that will be applied in this paper are described briefly. In section 5, we provide the proposed hybridization algorithm for clustering. Detailed experiments results are provided in section 6. Finally, Section 7 concludes the paper.

2 PREVIOUS WORK

Since a decade ago, the natural inspired algorithms that are based on the ACO meta-heuristic have been applied to many different problems, but research on optimizing ACO parameters has been sparse. There are many enhancement proposed for Ant-based clustering algorithm that are computationally demanding and require additional user-specified parameters, however there is no general guidelines are available. Many research efforts have gone in the past few years to find the best parameters was tackled by several researchers. Pilat *et al.* [25] and Gaertner [9] used the genetic algorithms to optimize the ACS algorithm parameters for the traveling sales persons (TSP) problem. Another paper that tackles the problem of parameter selection is [27], which uses an ACO algorithm to find the best ants which then use an ACO algorithm to find the best tour. The author of this work modified ACS in such a way that it evolves parameters based on an extra pheromone matrix maintained solely for this purpose. Guntsch and Middendorf in [12] describe a population based approach for ACO where all pheromone information corresponds to solutions that are members of the actual population. Another attempt to combine genetic algorithms with ideas from ACO was termed GAACO by Acan [3]. He uses both algorithms in parallel using the same problem representation for both and allowing solutions to migrate from one algorithm to another. Aranha and Iba [4], try to optimize ant-clustering algorithm parameters using genetic algorithms, each individual is represented by these configuration parameters. For each generation, they run the program once with each set of parameters, and take the fitness from each run.

3 ANT INSPIRED CLUSTERING ALGORITHMS

Recently, the natural inspired intelligence techniques as promising techniques have been applied to clustering optimization problems, *ant colony optimization* (ACO) [4, 5, 9, 21, 22]. These studies are often inspired by the observation of social insects and other animal societies [12, 25, 27]. These algorithms, mainly stochastic search and optimization techniques, are guided by the principles of *collective behavior* 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. The ACO [7, 8], which focuses on discrete optimization problems, have been used in clustering algorithms [14]. Deneubourg *et al.* [6] proposed an agent–based model to explain the clustering behavior of real ants. In this model, artificial ants (or agents) are moving randomly on a square grid of cells on which some items are scattered. Each cell can only contain a single item and each ant can move the items on the grid by picking up and dropping these items with a certain probability which depends on an estimation of the density of items of the same type in the neighborhood. Lumer and Faieta [23] extended the model of Deneubourg *et al.*, [6] using a dissimilarity–based evaluation of the local density, in order to make it suitable for data clustering. In their model, objects represent data items that belong to a database. These items are randomly scattered on a periodic square grid on which randomly moving agents group them according to their similarity. In order to do that, a similarity (or dissimilarity) measure between pairs of data items is needed to compute the probabilities of picking and dropping data elements on the grid. In their model, the probability of picking a data element x_i is defined as

$$P_p(x_i) = \left(\frac{k_p}{k_p + f(x_i)} \right)^2 \qquad (2)$$

where k_p is a constant and $f(x_i)$ is a similarity density measure with respect to element x_i. Likewise, the probability of dropping a data element is given by

$$P_d(x_i) = \begin{cases} 2f(x_i) & if\, f(x_i) < k_d \\ 1 & otherwise \end{cases} \qquad (3)$$

where k_d is a constant. The similarity density $f(x_i)$ for an element x_i, at a particular grid location τ, is defined as

$$f(x_i) = \max\left\{ \frac{1}{s^2} \sum_{x_j \in neigh(\tau)} \left(1 - \frac{d(x_i, x_j)}{\alpha} \right), 0 \right\} \qquad (4)$$

where s^2 is the size of the perception area $neigh(\tau)$, centered at the location of the agent and α is a scaling factor of the dissimilarity measure $d(x_i, x_j)$ between elements x_i and x_j. Generally, the size of the neighborhood is 3*x*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. By following these rules, objects that are near each other in the feature space will be likely to be dropped in neighboring positions in the work space. After an initial period of random activity, a small tentative cluster of few similar objects will form. This pre-cluster acts as a *stigmergic beacon* so that the probability of dropping new, similar objects near it is greater than anywhere else on the workspace. This leads to a positive feedback cycle which increases the size of the cluster, until the clustering process is complete. The ant-based clustering algorithms, as a general rule, can be considered as *non-hierarchical, hard, agglomerative* clustering methods as shown in Table 1. After the first appearance of this algorithm, many other variations of it have been proposed to improve its output quality [14, 16], its convergence speed [13, 15]. In a similar way, in [19, 20] 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.

TABLE 1. THE MAIN STEPS FOR BASIC ANT CLUSTERING ALGORITHM.

```
function Ant-Clustering (Ants, Data, Parameters) returns clusters
  inputs: Ants, a set of sorting and grouping ants
          Data, data items that needs to be clustered
          Parameters,
  randomly scatter data items on the toroidal grid
  loop for j from 1 to size (Ants) do
     i := random_select(free data items)
     Pick_up(Ants(j), i)
     g := random_select(empty grid locations)
     place_agent(Ants(j), g)
  end do
  repeat
     j := random_select(Ants)
     step(Ants(j), stepsize)
     i := carried_item(Ants(j))
     drop := drop_item?(f(i))        // According Eqs. (3) and (4)
     if drop = True then
        drop(Ants(j), i)
        pick := False
        while pick = False do
           i := random_select(free data items)
           pick := pick_item?(f(i))  // According Eqs.(2) and (4)
        end while
        pick_up(Ants(j), i)
     end if
  until all data items is clustered, or enough time has elapsed
return the data clusters
```

Later, Gu and Hall [11], proposed fuzzy ants clustering algorithm with a kernel distance metric reformulation and the goodness of the partition is evaluated by using the kernel Xie-Beni criterion after an epoch. Moreover, 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 [28]. Monmarche *et al.*, [24] proposed an algorithm where several objects are allowed to be on the same cell of the workspace grid. Each cell with one or more objects together corresponds to a cluster. Each ant is also capable of carrying more than one object at a time. In this way, a kind of hierarchical clustering is implemented, where an ant carries an entire heap of objects. Another contribution of Monmarche was to hybridize the ant-based clustering algorithm with *k*-means algorithm and compared it to traditional *k*-means on various data sets, using the classification error for evaluation purposes. Ramos *et al.* [26] proposed *ACLUSTER* algorithm, which modified the ant-based clustering by changing the movement paradigm. While the previous works all relied on random moving ants, his ants would move according to a trail of pheromones left on clustering formations. This would reduce the exploration of empty areas, where the pheromone would eventually evaporate. In that sense, bio-inspired spatial transition probabilities are incorporated into the system, avoiding randomly moving agents, which encourage the distributed algorithm to explore regions manifestly without interest. The strategy allows guiding ants to find clusters of objects in an adaptive way. Hartmann [17] tried a different approach to the ant clustering algorithm, by using evolution to train both the system's disparity function and move policy. Each ant would have a neural network which would take the objects of its vicinity as input, and return the move action, and the pick up or drop action, as outputs. By changing the evolutionary system fitness function.

4 PARTICLE SWARM OPTIMIZATION

The particle swarm optimization algorithms (PSO) are based on two socio-metric principles. Particles fly through the solution space and are influenced by both the best particle in the particle population and the best solution that a current particle has discovered so far. The best particle in the population is typically denoted by (gobal best), while the best position that has been visited by the current particle is donated by (local best). The (global best) individual conceptually connects all members of the population to one another. That is, each particle is influenced by the very best performance of any member in the entire population. The (local best) individual is conceptually seen as the ability for particles to remember past personal success. The particle swarm optimization makes use of a velocity vector to update the current position of each particle in the swarm. The position of each particle is updated based on the social behavior that a population of individuals adapts to its environment by returning to promising regions that were previously discovered [8]. Let the *i*-th particle of the swarm is represented by the *D*–dimensional vector $X_i = (x_{i1}, x_{i2}, ..., x_{iD})$ and the best particle in the swarm, i.e. the particle with the smallest function value, is denoted by the index *g*. The best previous position (the position giving the best function value) of the *i*-th particle is recorded and represented as $P_i = (p_{i1}, p_{i2}, ..., p_{iD})$, and the position change (velocity) of the *i*-th particle is $V_i = (v_{i1}, v_{i2}, ..., v_{iD})$. The particles are manipulated according to the equations

$$v_{id} = w \cdot v_{id} + c_1 \cdot r_1 \cdot (p_{id} - x_{id}) + c_2 \cdot r_2 \cdot (p_{gd} - x_{id}) \quad (5)$$

$$x_{id} = x_{id} + v_{id} \quad (6)$$

where $d = 1, 2, \ldots, D$; $i = 1, 2, \ldots, N$ and N is the size of population; w is the inertia weight; c_1 and c_2 are two positive constants; r_1 and r_2 are two random values in the range [0, 1]. The first equation is used to calculate ***i***-th particle's new velocity by taking into consideration three terms: the particle's previous velocity, the distance between the particle's best previous and current position, and, finally, the distance between swarm's best experience (the position of the best particle in the swarm) and ***i***-the particle's current position. Then, following the second equation, the ***i***-the particle flies toward a new position. In general, the performance of each particle is measured according to a predefined fitness function, which is problem–dependent. The role of the inertia weight w is considered very important in PSO convergence behavior. The inertia weight is employed to control the impact of the previous history of velocities on the current velocity. In this way, the parameter *w* regulates the trade–off between the global (wide–ranging) and local (nearby) exploration abilities of the swarm. A large inertia weight facilitates global exploration (searching new areas); while a small one tends to facilitate local exploration, i.e. fine–tuning the current search area. A suitable value for the inertia weight *w* usually provides balance between global and local exploration abilities and consequently a reduction on the number of iterations required to locate the optimum solution. A general rule of thumb suggests that it is better to initially set the inertia to a large value, in order to make better global exploration of the search space, and gradually decrease it to get more refined solutions, thus a time decreasing inertia weight value is used. The main steps of the PSO algorithm are shown in Table 2, where there are three main steps 1) initialize a population of particles (position and velocities); 2) updating velocities; 3) updating positions.

TABLE 2. THE MAIN STEPS FOR PSO ALGORITHM.

```
function PSO (Particles, Fitness-FN) returns a best particle
   inputs: Particles, a set of solutions
           Fitness-Fn, an optimization function according Eq. (7).
   repeat
      loop for i from 1 to size (Particles) do
          Find the personal-best & global-best position of Particles (i).
          update Particles (i) according to Eq. (5) & Eq. (6).
      end do
   until best particle is fit enough, or number of generation reached
return the best particle in Particles, according to Fitness-Fn
```

5 HYBRID MODEL

In the Ant-based clustering algorithms, there are large numbers of parameters and their sensitivities or its optimal values are not known. For example, the *size of the neighborhood* evaluated on the probability function, sometimes also called the sensory ability of the ant, has a noticeable impact on the algorithm performance. A small area improves the convergence time, but may generate less homogeneous clusters. A larger area improves the clustering quality, but makes the generation of the initial clusters more difficult. The *pick* and *drop* probability functions – These are the core of the ant-clustering algorithm. The disparity function $f(x_i)$ must vary according to the data which one wishes to partition, just like in other clustering algorithms. It can, however, be learned by a neural network or evolutionary heuristic. The *pick* and *drop* probability functions, which are based on the $f(x_i)$ disparity function, can be tailored to change the shape or the density of the resulting clusters. The move policy on grid square for each agent at each time step. It has been proposed that a more objective move policy would improve convergence times, by reducing the time the ants spend at places where there is no work to be done. However, it is suspected that by reducing randomness might also compromise the emergence factor of the system. Among some proposed move policies are movement by pheromones, where the ant leave pheromones near places where they picked up or dropped objects, to guide other ants to these "hot" spots, local memories, so that ants remember the position where they dropped objects of certain kinds, and teleporting to available objects, to speed up the picking a new object. However, there are a large number of parameters in the ant-clustering algorithm, which exact effects in the performance of the technique are not yet very well known. Small differences in the constants might result in large differences in the results, so that hand-tuning is a daunting task. We propose the use of particle swarm optimization algorithms to automatically choose the best parameters for a given task.

TABLE 3. THE MAIN STEPS FOR HYBRID PSO-ANT-CLUSTERING ALGORITHM.

```
function PSO-Ant-Cluster (Ants, Data, Fitness-FN) returns clusters
   inputs: Ants, a set of sorting and grouping ants
        Data, data items that needs to be clustered
        Particles, a set of sensitive parameters values
        Fitness-Fn, a performance function according Eq. (7).
  repeat
      loop for i from 1 to size (Particles) do
         clusters = Ant-Clustering (Ants, Data, Particles(i))
         Find the personal-best position of Particles (i) according to Eq. (7).
         Find the global-best position of Particles (i) according to Eq. (7).
         update Particle (i) according to Eq. (5) & Eq. (6).
      end do
   until best particle is fit enough, or number of generation reached
    clusters = Ant-Clustering (Ants, Data, best particle)
return clusters
```

TABLE 4. PARAMETERS FOR THE ANT-CLUSTERING

Parameter name	Range value
Scaling factor α	(0,…,1)
Pick probability (K_{pick})	(0,…,1)
Drop probability (K_{drop})	(0,…,1)
Sensory ability (*neighborhood*)	(0,…,10)

5.1. Methodology

In order to determine the best adoptions parameters for the clustering problem, and the relationship between the clustering evaluation and these adaptations, the PSO-Ant-Clustering algorithm allows the parameters controlling the clustering sensitivity by adapt as the PSO search algorithm proceeds. The proposed hybrid model consists of a set of particles, each of which has an ant-clustering parameters set. The ants use these parameters to guide its search for favorable dropping/picking locations. The main goal of this is to create clusters of higher quality. The PSO-Ant-Clustering algorithm can be described in pseudo code as showing in Table 3. To improve the ant clustering algorithm, we'll try to optimize its parameters presented in Table 4. However, there are several of the parameters can be set independently of the data. These include the number of ants, which we set to be 20, the size of the ants' short-term memory, and the number of iterations.

5.2. Objective Function

Simply comparing the number of clusters found in the embedding with the number of clusters contained in the original data can give an idea of the performance of the algorithm, although on a very superficial level. The correct number of clusters is known beforehand, being a parameter for the generation of each dataset prior to clustering. The PSO optimizes this pre-defined criterion or objective function. Objective functions which are commonly used as clustering criteria is the Pearson correlation coefficient depicted in Equation 7, which provides information on the degree of linear relationship between the distributions of two variables. In the context of clustering problem it can therefore be employed to determine the degree to which a mapping preserves a linear relationship between the distances in data items in data-space (described by distribution X) and their respective spatial distances on the grid (described by distribution Y)

$$P = \frac{\text{Covariance}(X, Y)}{\text{Variance}(X, Y) \cdot \text{Variancel}(X, Y)} \tag{7}$$

P takes values in the interval [−1, 1], with 1 signifying perfect positive correlation, −1 signifying perfect negative correlation, and 0 signifying a complete lack of linear correlation [14].

6 EXPERIMENTAL RESULTS

The proposed technique can be applied for an arbitrary number of dimensions by projecting the data sets into this 2-D space. However, for conceptual clarity the selected data sets were a two-dimensional data set identical to the one in Lumer and Faieta [23]. They have used a simple example where the attribute space is $\Re^2$, and the values of the two attributes for each object correspond to its coordinates (x, y) in $\Re^2$. Four clusters of 200 points each are generated in attribute space, with *x* and *y* distributed according to normal (or Gaussian) distributions. The experimental results comparing the PSO-Ant clustering algorithm with the traditional ant-clustering algorithm. For PSO, all swarm particles start at a random position in the range as shown in Table 4, for each dimension. The velocity of each particle is randomized to a small value to provide initial random impetus to the swarm. The swarm size was limited to 25 particles. The must important factor is *maximum velocity* parameter which affect the convergence speed of the algorithm is set to 0.5. the number of iterations is 50, and the c_1 and c_2 are 2.0 and 2.0 respectively. For the ant-clustering algorithm, the colony size is 20 and the number of iteration is 50000. T

TABLE 5: THE COMPARISON BETWEEN TRADITIONAL ANT-CLUSTERING AND HYBRID PSO-ANT-CLUSTERING TECHNIQUES

	Pearson correlation coefficient
PSO-Ant-Clustering	0.71
Ant-Clustering	0.59

For the average of the fitness of the PSO as shown in Figure 1, there is a increasing global tendency, and after iteration 10 we can see the average of the fitness is kept on a fixed range,

TABLE 6. THE PARAMETERS FOR THE ANT-CLUSTERING

Parameter name	Estimated values	Hand values
Scaling factor α	0.65	0.7
Pick probability (K_{pick})	0.065	0.1
Drop probability (K_{drop})	0.02	0.125
Sensory ability (*neighborhood*)	6	5

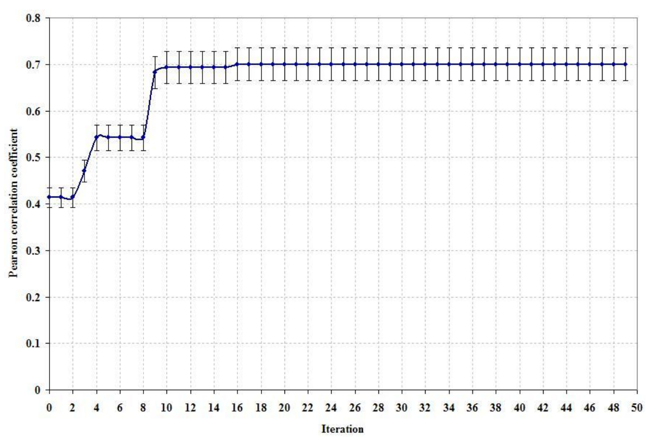

FIGURE 1. THE AVERAGE OF LEARNING RATE OF PSO-ANT-CLUSTERING ALGORITHM.

The algorithms are implemented using Java. Table 5 present a comparison among the results of PSO-Ant-Clustering for 100 run on the mentioned data sets. The Pearson correlation coefficient performance criteria were improved by 20% using PSO-Ant-Clustering. The best estimated parameters are shown in Table 6. Figures 2 and 3 shows snapshot of resulting cluster using the two techniques.

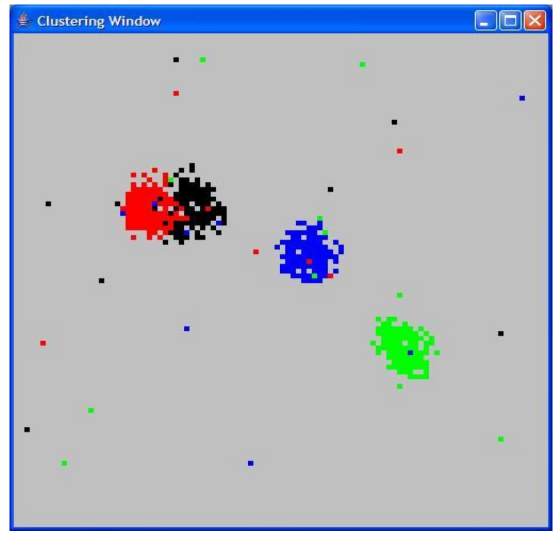

FIGURE 2. THE RESULTS OF BASIC ANT-CLUSTERING ALGORITHM.

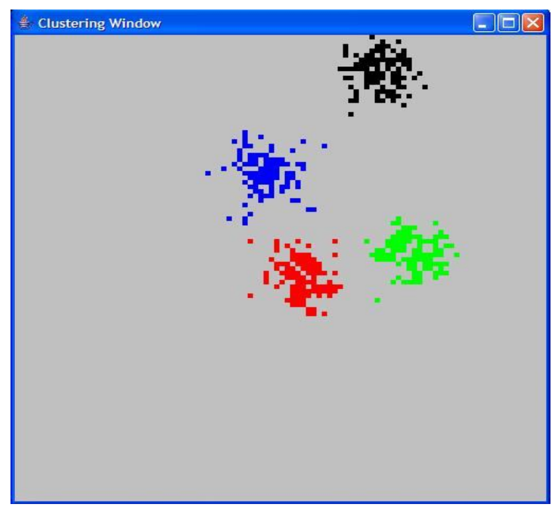

FIGURE 3. THE RESULTS OF HYBRID PSO-ANT-CLUSTERING ALGORITHM.

7 CONCLUSIONS AND FUTURE WORK

In the preceding sections the motivation for ant based clustering algorithms has been outlined together with its drawbacks with there are large numbers of parameters and their sensitivities or its optimal values are not known. Using the particle swarm optimization algorithm as another form of natural-inspired artificial intelligence technique can enhance the performance of ant-clustering algorithm by search the optimal parameters. However, in this study the only one data set is used in the future we plan to use some real data and other optimization function.

8 REFERENCES

[1]. Abraham A., Guo, H., and Liu H. Swarm Intelligence: Foundations, Perspectives and Applications, Studies in Computational Intelligence (SCI) 26, 3–25. Springer-Verlag, 2006.

[2]. Abraham A., Swagatam D., and Roy, S., Swarm Intelligence Algorithms for Data Clustering", in Soft Computing for Knowledge Discovery and Data Mining, Oded Maimon and Lior Rokach (Eds.), Springer-Verlag, Germany, 2008.

[3]. Acan A., "GAACO: A GA+ACO hybrid for faster and better search capability", In Ant Algorithms, pages 300–301, 2002.

[4]. Aranha C. and Iba H., "The effect of using evolutionary algorithms on Ant Clustering Techniques", ASGP 2006.

[5]. Bonabeau, E., Dorigo, M., and Theraulaz, G., Swarm Intelligence from Natural to Artificial Systems, Oxford University Press, 1999.

[6]. Deneubourg, L., Goss, S., Franks, N., Sendova–Franks, A., Detrain, C., Chrétien, L., "The Dynamics of Collective Sorting Robot–Like Ants and Ant–Like Robots", From Animals to Animats: Proc. of the 1st Int. Conf. on Simulation of Adaptive Behavior, 356–363, 1990.

[7]. Dorigo M, Maniezzo V and Colorni A, "The Ant System: Optimization by a Colony of Cooperating Agents", IEEE Trans. Systems Man and Cybernetics -Part B, vol. 26, 1996.

[8]. Dorigo M., and Stutzle T., Ant Colony Optimization. MIT Press, Cambridge, MA, 2004.

[9]. Gaertner D., Natural algorithms for optimization problems. Masters thesis, Imperial College London, 2004.

[10]. Grosan, C., Abraham, A., and Chis, M., Swarm Intelligence in Data Mining, Studies in Computational Intelligence (SCI) 34, 1–20. Springer-Verlag, 2006.

[11]. Gu Y., and Hall L., "Kernel Based Fuzzy Ant Clustering with Partition Validity". In IEEE International Conference on Fuzzy Systems, 2006.

[12]. Guntsch M., and Middendorf M., "A Population Based Approach for ACO", In Proceedings of EvoWorkshops2002, volume 2279, pages 71–80, Kinsale, Ireland, 3-4. Springer-Verlag, 2002.

[13]. Handl J., and Meyer B., "Improved Ant-based Clustering and Sorting in a Document Retrieval Interface". In Proceedings of the Seventh International Conference on Parallel Problem Solving from Nature, pages 913–923, 2002.

[14]. Handl J., Knowles J., and Dorigo M., "Ant-Based Clustering and Topographic Mapping", Artificial Life, 11(2), 2005.

[15]. Handl J., Knowles J., and Dorigo M., "On the Performance of Ant-Based Clustering". Frontiers in Artificial Intelligence and Applications, Vol. 104, pp. 204-213, 2003.

[16]. Handl J., Knowles J., and Dorigo M., "Strategies for the Increased Robustness of Ant-based Clustering". Lecture Notes in Computer Science, Vol. 2977, pp. 90-104, 2004.

[17]. Hartmann V., "Evolving Agent Swarms for Clustering and Sorting". In Genetic Evolutionary Computation Conference, GECCO, Vol. 1, pp.217-224, 2005.

[18]. Jain, A., Murty, M., and Flynn P., "Data Clustering: a Review", ACM Computing Surveys, Vol. 31, No. 3, pp. 264–323, 1999.

[19]. Kanade P., and Hall L., "Fuzzy Ant Clustering by Centroid Positioning". In Proceedings of IEEE International Conference on Fuzzy Systems, Hungary, pp. 371–376. 2004.

[20]. Kanade P., and Hall L., "Fuzzy Ants as a Clustering Concept", Proc. of the 22nd Int. Conf. of the North American Fuzzy Information Processing Society. 227-232, 2003.

[21]. Kennedy J., and Eberhart R., "Particle Swarm Optimization", In Proceedings of IEEE International conference on Neural Networks, pp. 1942-1948, 1995.

[22]. Kennedy, J., Eberhart, R., and Shi, Y., Swarm Intelligence, Morgan Kaufmann Publishers, Inc., San Francisco, CA, 2001.

[23]. Lumer E., and B. Faieta B., "Diversity and Adaptation in Populations of Clustering Ants". In Proc. O f the Third International Conference on the Simulation of Adaptive Behavior: From Animals to Animats 3, pp. 449-508. 1994.

[24]. Monmarche N., Slimane M., and Venturini G., "On Improving Clustering in Numerical Databases with Artificial Ants". Advances in Artificial Life, pp. 626-635, 1999.

[25]. Pilat M., and White T., "Using Genetic Algorithms to Optimize ACS-TSP", In ANTS '02: Proceedings of the Third International Workshop on Ant Algorithms, pages 282–287. Springer-Verlag, 2002.

[26]. Ramos V., Muge F., and Pina P., "Self-Organized Data and Image Retrieval as a Consequence of Inter-Dynamic Synergistic Relationships in Artificial Ant Colonies", In Proceedings of the 2nd International Conference on Hybrid Intelligent Systems, 500-509, 2002.

[27]. Randall M., "Near parameter Free Ant Colony Optimization", In ANTS Workshop, pages 374–381, 2004.

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