INTERNATIONAL JOURNAL of MATHEMATICS AND COMPUTERS IN SIMULATION

Dynamically Adaptive Data Clustering Using Intelligent Swarm-like Agents

Sherin M. Youssef, Mohamed Rizk, Mohamed El-Sherif

Abstract — Inspired by the self-organized behaviour of bird flocks, a new dynamic clustering approach based on Particle Swarm Optimization is proposed. This paper introduces a novel clustering method, the PSDC, a new Particle Swarm-like agents approach for Dynamically Adaptive data clustering. Unlike other partition clustering algorithms, this technique does not require initial partitioned seeds and it can dynamically adapt to the changes in the global shape or size of the clusters. In this technique, the agents have lots of useful features such as sensing, thinking, making decisions, parallelism and moving freely in the solution space. The moving swarm-like agents are guided to move according to a specific proposed navigation rules. These rules help every agent to find its new position in its navigation process and the clustering results emerge from the collective and cooperative behaviour of these swarm agents. If the swarm performance showed gradual improvements during a predefined number of cycles, then the current population could pass useful information to the next population in order to help further generations in reaching better solutions faster and enable the learning process to be reinforced. The distributed, adaptive and cooperative behaviour of these agents was so powerful to explore the solution space effectively. Through the cooperative behaviour, the generations of agents were able to build knowledge and the whole population could pass information to the next generation. Numerous experiments have been conducted using both synthetic and real datasets to evaluate the efficiency of the proposed model. Cluster validity approaches are used to quantitatively evaluate the results of the clustering algorithm. Experimental results showed that the proposed particle swarm-like clustering algorithm reaches good clustering solutions and achieves superior performance compared to others.

Keywords — Agents, clustering, Ant clustering, k-means.

I. INTRODUCTION

Clustering [1, 2] is a powerful, popular tool for discovering structure in data. Clustering analysis, which is the subject of active research in several fields such as statistics, pattern recognition, machine learning, and data mining, is to partition a given set of data or objects into clusters. It also has been applied in a large variety of applications, for example, image segmentation, object and character recognition, document retrieval, etc.

Manuscript received January 5, 2007; Revised Version: Febr.28, 2007

S. M. Youssef is with the Computer Engineering Department, College of Engineering & Technology, Arab Academy for Science and Technology, Alexandria, Egypt. (email: sherin@aast.edu)

M. Rizk is with the Electrical and communication Engineering Department, Faculty of Engineering, University of Alexandria, Alexandria, Egypt. (email: mrmrizk@ieee.org)

M. El-Sherif is with Regional Informatics Center, Arab Academy for Science and Technology (AAST), Alexandria, Egypt. (email: elsherif@aast.edu).

Data clustering is the process of identifying natural groupings or clusters, within multidimensional data, based on some similarity measures [2, 3]. Classical algorithms are static, centralized, and batch. They are static because they assume that the data and similarity function do not change while clustering is taking place. They are centralized because they rely on data structures (such as similarity matrices) that must be accessed, and sometimes modified, at each step of the operation. They are batch because they run their course and then stop. It is the unsupervised classification of patterns (observations, data items, or feature vectors) into groups (clusters). The clustering problem has been addressed in many contexts and by researchers in many disciplines; this reflects its broad appeal and usefulness as one of the steps in exploratory data analysis. However, clustering is a difficult problem combinatorially, and differences in assumptions and contexts in different communities have made the transfer of useful generic concepts and methodologies slow to occur [2].
A cluster is usually identified by a cluster center (or *centroid*). Data clustering is a difficult problem as the clusters in data may have different shapes and sizes. Furthermore, it is usually not known how many clusters should be formed. Most clustering algorithms are based on two popular techniques known as hierarchical and partitional clustering [2]. In hierarchical clustering, the output is "a tree showing a sequence of clustering with each clustering being a partition of the data set". Such algorithms have the following advantages [1]:

- The number of clusters need not be specified *a priori*, and
- They are independent of the initial conditions.

However, hierarchical clustering techniques suffer from the following drawbacks:

- They are static, i.e. data points assigned to a cluster cannot move to another cluster.
- They may fail to separate overlapping clusters due to a lack of information about the global shape or size of the clusters.

On the other hand, partitional clustering [1] algorithms partition the data set into a specified number of clusters. These algorithms try to minimize certain criteria (e.g. a square error function) and can therefore be treated as optimization problems. The advantages of hierarchical algorithms are the disadvantages of the partitional algorithms and vice versa.

Partitional clustering techniques are more popular than hierarchical techniques in pattern recognition, hence, this paper will concentrate on partitional techniques. Several researchers have applied the biological algorithm to engineered systems. These implementations fall into two broad categories: those in which the digital ants are distinct from the data objects being clustered, and those that eliminate this distinction. All of these examples form a partition of the data objects, without any hierarchical structure [4].

Particle swarm optimizers (PSO) [1] are population-based optimization algorithms modelled after the simulation of social behaviour of bird flocks. PSO is generally considered to be an evolutionary computation (EC) paradigm. Other EC paradigms include genetic algorithms (GA), genetic programming (GP), evolutionary strategies (ES), and evolutionary programming (EP). These approaches simulate biological evolution and are population-based. In a PSO system, a swarm of individuals (called *particles*) fly through the search space. Each particle represents a candidate solution to the optimization problem. The position of a particle is influenced by the best position visited by itself (i.e. its own experience) and the position of the best particle in its neighbourhood (i.e. the experience of neighbouring particles). When the neighbourhood of a particle is the entire swarm, the best position in the neighbourhood is referred to as the global best particle, and the resulting algorithm is referred to as a *gbest* PSO. When smaller neighbourhoods are used, the algorithm is generally referred to as *lbest* PSO. The performance of each particle (i.e. how close the particle is from the global optimum) is measured using a fitness function that varies depending on the optimization problem.

PSO was originally developed by Eberhart and Kennedy in 1995 [5], and was inspired by the social behaviour of a flock of birds. In the PSO algorithm, the birds in a flock are symbolically represented as particles. These particles can be considered as simple agents "flying" through a problem space. A particle's location in the multi-dimensional problem space represents one solution for the problem. When a particle moves to a new location, a different problem solution is generated. This solution is evaluated by performance indicators that provide a quantitative value of the solution's utility. In this paper a new approach of data clustering using particle swarm-like agents will be introduced. In section 3 the PSDC model will be discussed.

The remainder of this paper is organized as follows. section II presents some related work. In section III, an overview of ant clustering is introduced with an illustration of the modified ant-clustering algorithm. In section IV, we introduce the proposed Particle Swarm-Based Data Clustering (PSDC) algorithm with an illustration of the model description, the algorithm description, and agent decision rules and a brief pseudocode is illustrated. Experimental results are introduced in section V and, finally, conclusions are presented in section VI.

II. RELATED WORK

In recent years, it has been recognized that the partitioning techniques is well suited for clustering a large datasets due to their relatively low computational requirements. The time complexity of the partitioning technique is almost linear, which makes it widely used. The well-known partitioning algorithm is the K-means algorithm [2, 6, 7] and its variants. This algorithm is simple, straightforward and is based on the firm foundation of analysis of variances. The K-means algorithm clusters a group of data vectors into a predefined number of clusters. It starts with a random initial cluster center and keeps reassigning the data objects in the dataset to cluster centers based on the similarity between the data object and the cluster center. The reassignment procedure will not stop until a convergence criterion is met (e.g., the fixed iteration number or the cluster result does not change after a certain number of iterations). The main drawback of the K-means algorithm is that the cluster result is sensitive to the selection of the initial cluster centroids and may converge to the local optima. Therefore, the initial selection of the cluster centroids decides the main processing of K-means and the partitioning result of the dataset as well. Another limitation of the K-means algorithm is that it generally requires a prior knowledge of the probable number of clusters for a data collection. To deal with the limitations that exist in traditional partition clustering methods a number of computer scientists in recent years have proposed several approaches [3] inspired from biological collective behaviors to solve the clustering problem, such as Genetic Algorithm (GA) [8], Particle Swarm Optimization (PSO), Ant clustering and Self-Organizing Maps (SOM) [9]. Within these clustering algorithms, Ant clustering algorithm is a partitioning algorithm that does not require a prior knowledge of the probable number to clusters or the initial partition. The Ant clustering algorithm was inspired by the clustering of corpses and eggs observed in the real ant colony. Deneubourg et al. proposed a ''Basic Model'' to explain the ants' behavior of piling corpses and eggs. In their study, a population of ant-like agents randomly moved in a 2D grid. Each agent only follows one sample rule: randomly moving in the grid and establishing a probability of picking up the data object it meets if it is free of load or establishing a probability of dropping down the data object if it is loading the data object. After several iterations, a clustering result emerges from the collective activities of these agents. Lumer, Faieta and other researchers extended this ''Basic Model'' and applied it to numerical data analysis. Wu and Handl proposed the use of Ant clustering algorithms for document clustering and declared that the clustering results from their experiments are much better than that from K-means algorithm. However, in Ant clustering algorithm, the clustered data objects do not have mobility themselves. The data objects' movements have to be implemented through the movements of a small number of ant agents, which will slow down the clustering speed. Since each ant agent that is carrying an isolated data object does not communicate with other ant agents, it does not know the best location to drop the data object. The ant agent has to move or jump randomly in the grid space until it finds a place that satisfies its data object dropping criteria, which usually consumes a large amount of computation time.

Among the many bio-inspired techniques, ant-based clustering algorithms have received special attention from the community over the past few years for two main reasons. First, they are particularly suitable to perform exploratory data analysis and, second, they still require much investigation to improve performance, stability, convergence, and other key features that would make such algorithms mature tools for diverse applications. In [10] the Ant Clustering algorithm, have been explained, defined its key components, the main branches in its field of research, and its perceived characteristics.

In [11], a number of modifications have been introduced that improve the quality of the clustering and, in particular in assigning short memory to each agent so it remembers the last few carried data items and their respective dropping positions, increasing radius of perception of agents to employ larger neighbourhoods in order to improve the quality of the clustering and sorting on the grid, spatial separation of clusters in order for individual clusters to be well-defined, settings parameters to be a set as a function of the size of the data set. Unlike other static and centralized clustering techniques, our proposed model can dynamically adapt to the changes and does not require a prior knowledge of the number of clusters in the datasets. It is more adaptive towards problems with dynamic changed information.

III. ANT IN CLUSTERING ANALYSIS

A. Overview of Ant Clustering

Applying ant colony system in clustering analysis is still a novel research area. Tsai and his colleges [12] employed the ant system with differently favourable strategy for data clustering. It is named ant colony optimization with different favour (ACODF). ACODF algorithm has the following desirable strategies. It first uses differently favourable ants to solve the clustering problem. Then, the proposed ant colony system adopts simulating annealing concept for ants to decreasingly visit the amount of cities and get the local optimal solutions. Finally, it utilizes tournament selection strategy to choose a path. Every ant only needs to visit few cities instead of all cities. Thus, the ant will reduce visiting the cities every iteration. After several iterations, the trail intensity close between nodes of trails will be increased. On the other hand, the trail intensity far between nodes of trails will be decreases. Therefore, ants will favour to visit the close nodes and then reinforcing the trail with their own pheromone. Finally, a number of clusters will be built.

In [13], they applied the ant colony system (ACS) for clustering problem. Based on ACS, it treats the data (objects or elements) as the ants. Thus, each ant has different properties. Basically, the process of data clustering is the process of ant looking for food.

The basic ant algorithm starts with an initialization phase, in which (i) all data items are randomly scattered on the grid, (ii) each agent randomly picks up one data item; and (iii) each agent is placed at random position on the grid. Subsequently, the sorting phase starts, in which (i) one agent is randomly selected; (ii) the agent performs a step of a given *stepsize* (in a randomly determined direction); and (iii) the agent probabilistically decides whether to drop its data item. In a case of "drop-decision", the agent drops the data item at its current grid position if this grid cell is not occupied by another data item, or in the immediate neighbourhood of it (it locates a nearby free grid cell by means of a random search). It then immediately searches for a new data item to pick up. This is done using an index that stores the positions of all "free" data items on the grid; the agent randomly selects one data item *i* out of the index, proceeds to its position on the grid, evaluates the neighbourhood function *f* and probabilistically decides whether to pick up the data item. It continues this search until a successful picking operation occurs. Only then the loop is repeated with another agent. A number of modifications have been introduced that improve the quality of the clustering and, in particular, the spatial separation between clusters on the grid, which is essential for the scheme of cluster retrieval.

B. A Modified Ant-Clustering Algorithm

In the basic ant algorithm [11], the agent performs a step of a given stepsize in a randomly determined direction on the grid. This consumes more time for the algorithm to converge. In this paper, a modification to the basic algorithm is dedicated to bias the agent's movement towards the right solution preventing them from moving away from the solution. Instead of moving randomly, the swarm agent senses the neighbourhood and a decision is made to move to one of its free neighbours in case of popping operation. Two operations have been defined; the "Popping" operation and the "Pushing" operation.

The behavior of an agent through the clustering operation is as follows: If the agent is not holding any object, it tries to pop one of the available objects, one at a time with probability given by equation 1 (Popping operation). Once it decides to pop up one object, the agent is moved into that position. Then it starts a random walk under some constraints over the workspace, looking for a place with high local similarity to push down the object with probability given by equation 2 (Pushing operation).

The following equations have been used and represent the main operations in the algorithm:

$$P^{pop} = \left(\frac{k_p}{k_p + f(i)} \right)^2 \qquad (1)$$

$$P^{push} = \left(\frac{f(i)}{k_d + f(i)} \right)^2 \qquad (2)$$

Equation 1 gives the probability of popping up an object. kp is a pop up threshold parameter. Equation 2 gives the probability of pushing down an object. kp is a drop threshold parameter.

$$f(i) = \frac{\sum M_d - d(i, x_s)}{S_t} \qquad (3)$$

Where x^s is an object within the neighbourhood radius of i, M^d is the maximum distance between any two objects, and S^t is the total number of objects in the neighborhood of i. We then calculate the crowdness factor, c(i) by the following equation shown below:

$$c(i) = \frac{S_t^2}{S_t^2 + k_{crowd}^2} \qquad (4)$$

The pseudo-code of the mechanism can be illustrated as shown below in Fig. 1.

```
BEGIN
Initialization phase
For every agent do
   - Select random item
   - Pop up the item to the agent
   - Place the busy agent in a free location randomly
End for
For counter=1 to maximum No. of iterations do
        -  Select random agent
        -  Move the agent a unit step according to a
           defined constraints (within a range towards the
           neighbourhood)
        -  If it is possible to push down the item according
           to (eq. 2) then
              While (no pop up for an item) do
                    Select a free item randomly (within
              a range)
                    Pop up the item according to (eq.
              1)
              End while
          End if
End for
END
```

Fig. 1 A brief pseudo-code of a modified Ant-Clustering Algorithm

IV. THE PROPOSED PARTICLE SWARM-BASED CLUSTERING ALGORITHM (PSDC)

This section describes the proposed model and presents the phases of the proposed technique.

A. Model Description

The dataset to be clustered are represented as a set of vectors $X = \{x_1, x_2, x_3, ..., x_N\}$ where the vector x_i corresponds to a single data object in the D-dimension space and x_i is the feature vector of the i^{th} data object. N represents the total number of data objects in the dataset. In the proposed model, a population P of swarm agents is modelled as a set of agents $S = \{s_1, s_2, s_3, ..., s_M\}$ where M is the total number of agents, where $M \leq N$. Each agent is modelled as a moving object s_i of a circular sensing range of radius r_i, it has the ability to move freely in the solution space according to the proposed agents' rules (as will be explained in next subsection). Each independent agent can cooperate with its competitors in the neighbourhood to accelerate the discovery of the right clusters. More than one agent can share the same place or data object without any collision or fighting. In addition none of the agents could be lost in the space.

The similarity between two data objects needs to be measured in clustering analysis. In order to group similar data objects, proximity metric has to be used to identify data objects that are similar. Over the years, two prominent ways have been proposed to compute the similarity between data objects. The first method is the Euclidean distance [2, 3], the other one is the cosine correlation measure [3]. The last method is used in case of real data. In the proposed mechanism, one population passes information to the next one depending on how fitted clustering solutions it reached in their past navigation. (as will be explained in next subsection).

B. Algorithm Description

The proposed PSDC algorithm passes through the following phases.

B.1 Starting phase

M agents are randomly generated in the solution space where $M \leq N$ and N is the total number of data objects. Every agent is assigned to a data object randomly to be dedicated to it. The sensing range also must be initialized.

B.2 Decision phase

Every agent in the solution space moves according to guiding rules from which it decides the new updated position for it. The PSDC model consists of three simple steering rules based on the flocking model rules [3]. These rules need to be executed at each instance over time, for each individual agent. These basic rules are listed below.

Rule 1: Separation, steering to avoid collision with other boids nearby. The separation rule acts as an active boid trying to pull away before crashing into each other.

$$\Pi = (s_n - s_a) / \text{dist}(s_n, s_a) \qquad (5)$$

Where s_n is the position of agent's neighbor, s_a is the current position of the agent, dist (s_n, s_a) is the distance between the current agent and the selected neighbor.

Rule 2: Alignment, steering toward the average heading and speed of the neighboring flock mates. The alignment rule, acts as the active boid trying to align its vector with the average vector of the flock in its local neighborhood. The degree of locality of this rule is determined by the sensor range of the active flock boid.

$$\mu = (s_n - s_a) / G \qquad (6)$$

Where s_n is the position of agent's neighbor, s_a is the current position of the agent, G is the total number of neighbors within the current agent's sensor range.

Rule 3: Cohesion, steering to the average position of the neighboring flock mates. The cohesion rule acts as an active boid trying to orient its vector in the direction of the centroid (average spatial position) of the local flock.

$$\beta = (s_n - s_a) \qquad (7)$$

Where s_n is the position of agent's neighbor, s_a is the current position of the agent. We found in case of geometric datasets the separation rule is not suitable, but in the distributed form it is recommended. To achieve comprehensive flocking behavior, the actions of all the rules are summed to give a net vector required for the active flock boid producing a new position for the agent.

B.3 Movement phase

In this phase the agent updates its position by adding a delta value to its current position. The delta value is measured as follows:

$$V_{net} = \Pi + \mu + ß. \qquad (8)$$

Where the parameters Π, μ and ß were calculated in equations 5, 6 and 7 respectively.

After calculating the V_{net} then the agent is able to find its new position for the next iteration using the following equation

$$s_{new} = s_{current} + V_{net} \qquad (9)$$

The previous scenario will be repeated within the same cycle for all agents until convergence occurs. The population decides to stop navigating in the solution space if the maximum difference in change in its position is less than a defined threshold, this means that all the agents are stable and almost stopped moving. The fitness value is measured every complete cycle to decide if the population made a good solution or a new population must be generated to get better results. The convergence obtained if the fitness measure has the same value for a number of cycles without any change, the convergence window is set to 20 cycles.

In case of stigmergy, useful information is passed to the next population in the form of virtual points added to the input datasets to helps the new population to find the right clusters in less time and with better performance results. The virtual points are the positions of old centers of the last discovered clusters.

B.4 Checking New Clusters Solution ( )

The previous agents iteration-phases are repeated until approximately no agent movements are detected or a maximum number of iterations are reached for one population. Correspondingly, groups of agents will appear to be accumulated on centroids of the clusters of a new predicted solution of the i^{th} generation.

B.5 Passing Information Process ( )

If the performance indicators' values showed through the iterations gradual improvements during a predefined number of cycles, then the current population could pass useful clustering information to the next population in order to help further generations in reaching better solutions faster and reinforce learning process. The centroids of good clusters solutions will be reinforced by biased virtual points proportional to the fitness of previous solutions. This results in overall performance improvement and supports the cooperative and the adaptive features of the agents.

According to the previous section, the pseudocode of the proposed PSDC mechanism can be summarized as shown below in Fig. 2.

```
Parameters' Initialization
Repeat
  - Generate new population of agents.
  Repeat
    For every agent
      - Find agent's neighbors (data points)
        within its range.
      - Apply agent thinks and decision rules ( ).
      - Calculate the agent's new (position) according
        to position equations.
    End
    - Increment iteration number.
    - Check for a new clusters solution for the current
      Population ( ).
  - Evaluate the performance indicators ( ).
  - Pass useful information to next generation ( ).
  - Increment Cycle number.

Until convergence criteria is met.
```

Fig. 2 A brief pseudo-code of the proposed PSDC mechanism

V. EXPERIMENTAL RESULTS

Experiments have been carried out to validate the efficiency of the proposed model. In this section we illustrate the characteristics of the proposed technique with several artificial and real data sets. The performance of the PSDC algorithm has been compared with both the modified ant clustering algorithm and the k-means algorithm [10]. Moreover, the performance of the PSDC will be compared with the hierarchical agglomerative average-link clustering and the one-dimensional self-organising maps applied on three real data sets. All the experiments were run on Pentium 3.2 GHZ processor with 1 GB RAM. Experiments have bee conducted on various synthetic and real multidimensional data sets containing different number of clusters. Results are compared using different validity indices [14, 15, 16, 17] (as will be defined in the next secion). Comparative results are presented for synthetic datasets and two real data collections taken from the Machine Learning Repository [18].

A. Performance Indicators

The main subject of cluster validation [14] is "the evaluation of clustering results to find the partitioning that best fits the underlying data". Hence, cluster validity approaches are approaches used to quantitatively evaluate the result of a clustering algorithm. These approaches have representative indices, called validity indices [14]. The traditional approach to determine the "optimum" number of clusters is to run the algorithm repetitively using different input values and select the partitioning of data resulting in the best validity measure. Two criteria that have been widely considered sufficient in measuring the quality of partitioning a data set into a number of clusters:

- *Compactness*: samples in one cluster should be similar to each other and different from samples in other clusters. An example of this would be the variance of a cluster.
- *Separation*: clusters should be well-separated from each other. An example of this criterion is the Euclidean distance between the centroids of clusters.

In this section several validity indices are introduced. These indices are used for measuring "goodness" of a clustering result comparing to other ones which were created by other clustering algorithms, or by the same algorithms but using different parameter values.

A.1 The Dunn Index

The Dunn Index [14, 15, 16, 17] determines the minimal ratio between cluster diameter and inter cluster distance for a given partitioning. Thus, it captures the notion that, in a good clustering solution, data elements within one cluster should be much closer than those within different clusters. It is defined as

$$D = \min_{c,d \in C} \left[\frac{\delta(\mu_c, \mu_d)}{\max_{e \in C} [diam(e)]} \right], \qquad (10)$$

Where the diameter diam(*c*) of a cluster *c* is computed as the maximum inner cluster distance, and $\delta(u_c, u_d)$ is the distance between the centroids of clusters *c* and *d*. It is to be maximized. If the dataset contains compact and well-separated clusters, the distance between the clusters is expected to be large and the diameter of the clusters is expected to be small. Thus, based on the Dunn's index definition, we may conclude that large values of the index indicate the presence of compact and well-separated clusters.

A.2 The Davies-Bouldin Index

Davies and Bouldin [14, 15, 16, 17] proposed the following index that is known as Davies-Bouldin measure. It is a function of the ratio of the sum of within-cluster scatter to between-cluster separation.

Let *C = {C1, . . . ,Ck}* be a clustering of a set *D* of data objects.

$$DB = \frac{1}{k} \cdot \sum_{i=1}^{k} R_i, \qquad (11)$$ with

$$R_i = \max_{\substack{j=1,\ldots,n_c \\ i \neq j}} R_{ij} \quad \text{and} \quad R_{ij} = \frac{(s(C_i) + s(C_j))}{\delta(C_i, C_j)}, \qquad (12)$$

where $s : C \rightarrow \mathbf{R}$ measures the scatter within a cluster, and $\delta : C \times C \rightarrow \mathbf{R}$ is a cluster to cluster distance measure.

Given the centroids c_i of the clusters C_i, a typical scatter measure is

$$s(C_i) = \frac{1}{|C_i|} \sum_{x \in C_i} ||x - c_i||. \qquad (13)$$

and a typical cluster to cluster distance measure is the distance between the centroids, $\|c_i - c_j\|$. Because a low scatter and a high distance between clusters lead to low values of R_{ij}, a minimization of DB is desired.

The Dunn index and the Davies-Bouldin index are related in that they have a geometric (typically centroidic) view on the clustering. The measures work well if the underlying data contains clusters of spherical form, but they are susceptible to data where this condition does not hold.

A.3 The Inner Cluster Variance

The Inner Cluster Variance [11, 17, 19] computes the sum of squared deviations between all data items and their associated cluster centre, which reflects the common agreement that data elements within individual clusters must be similar. It is given by

$$I = \sum_{c \in C} \sum_{i \in c} \delta(i, \mu_c)^2, \qquad (14)$$

Where C is the set of all clusters, u_c is the centroid of cluster c and $\delta(i, u_c)$ is the distance function employed to compute the deviation between each data item *i* and its associated cluster centre. It is to be minimised.

A.4 The SD validity Index

The bases of SD validity index [14] are the average scattering of clusters and total separation of clusters. The scattering is calculated by variance of the clusters and variance of the dataset, thus it can measure the homogeneity and compactness of the clusters. The variance of the dataset and variance of a cluster are defined in Equation

Variance of the dataset:

$$\sigma_x^p = \frac{1}{n} \sum_{k=1}^{n} \left(x_k^p - \overline{x^p} \right)^2$$

$$\sigma(x) = \begin{bmatrix} \sigma_x^1 \\ \vdots \\ \sigma_x^d \end{bmatrix}$$

Variance of a cluster:

$$\sigma_{v_i}^p = \frac{1}{\|c_i\|} \sum_{k=1}^{n} \left(x_k^p - \overline{v_i^p} \right) \qquad (15)$$

$$\sigma(v_i) = \begin{bmatrix} \sigma_{v_i}^1 \\ \vdots \\ \sigma_{v_i}^d \end{bmatrix}$$

The average scattering for clusters is defined as:

$$Scatt = \frac{1}{n_c} \sum_{i=1}^{n_c} \frac{\|\sigma(v_i)\|}{\|\sigma(x)\|} \qquad (16)$$

The total separation of clusters is based on the distance of cluster centre points thus it can measures the separation of clusters. Its definition is given by Equation

$$Dis = \frac{\max_{i,j=1\ldots n_c} \left(\|v_j - v_i\| \right)}{\min_{i,j=1\ldots n_c} \left(\|v_j - v_i\| \right)} \sum_{k=1}^{n_c} \left(\sum_{\substack{j=1, \\ i \neq j}}^{n_c} \|v_j - v_i\| \right)^{-1} \qquad (17)$$

The SD index can be defined based on previous equations as follows

$$SD = \alpha \cdot Scatt + Dis \qquad (18)$$

where α is a weighting factor that is equal to Dis parameter in case of maximum number of clusters. Lower SD index means better cluster configuration as in this case the clusters are compacts and separated.

A.5 Fitness Measure

One particle in the swarm represents one possible solution for clustering the document collection. Therefore, a swarm represents a number of candidate clustering solutions for the document collection. Each particle maintains a matrix ***Xi*** =

$(C_1, C_2, \ldots, C_i, .., C_k)$, where C_i represents the i^{th} cluster centroid vector and k is the number of clusters. According to its own experience and those of its neighbors, the particle adjusts the centroid vector' position in the vector space at each generation. The average distance of data objects to the cluster centroid is used as the fitness value [5] to evaluate the solution represented by each particle. The fitness value is measured by the equation below:

$$f = \frac{\sum_{i=1}^{N_c} \{ \frac{\sum_{j=1}^{p_i} d(o_i, m_{ij})}{p_i} \}}{N_c} \qquad (19)$$

where m_{ij} denotes the j^{th} document vector, which belongs to cluster i; O_i is the centroid vector of the ith cluster; $d(o_i, m_{ij})$ is the distance between document m_{ij} and the cluster centroid O_i; P_i stands for the number of documents, which belongs to cluster C_i; and N_c stands for the number of clusters.

B. Using Syntactic Data Sets

Data generator program has been implemented to generate datasets of different sizes and different distributions. The tables and figures below illustrate some examples of generated datasets showing comparisons among the PSDC, K-means and Modified Ant algorithms in the performance indicators. Different generated datasets of sizes: 1000, 1800, 2800, 3200 and 3600 with different distributions and different number of clusters: 5, 9, 7, 8 and 9 respectively are used for comparisons. We used φ to represent the number of clusters, N to represent the number of the datasets, and *itr* to represent the number of` iterations for each algorithm.

TABLE 1
CPMPARING FITNESS'S VALUES FOR PSDC TECHNIQUE, K-MEANS AND MODIFIED ANT CLUSTERING ALGORITHM, FOR DATA SETS OF DIFFERENT SIZES AND DIFFERENT NUMBER OF CLUSTERS.

Fitness measure							
		PSDC		Kmeans		Ant	
N	φ	itr	Index value	itr	Index value	itr	Index Value
1000	5	2	7.699	6	8.3131	10	7.7311
1800	9	2	7.6075	12	8.778	10	7.6328
2800	7	2	7.6408	12	8.861	10	7.7199
3200	8	2	7.6353	10	9.046	10	7.7134
3600	9	2	7.5968	14	8.769	10	7.788

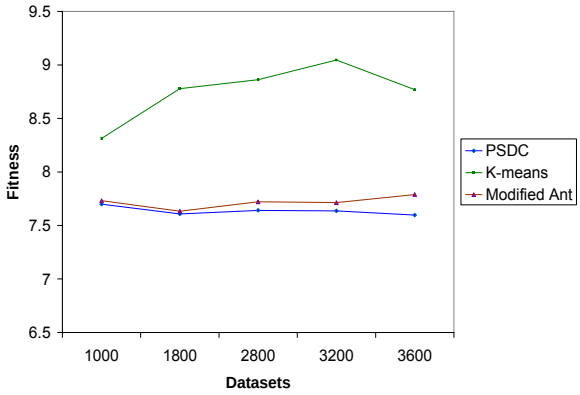

Fig. 3. Comparing fitness indicator results for PSDC technique, K-means and Modified Ant algorithm for experimental datasets of different sizes and distributions.

As observed from table 1 and Fig. 3, the values of the fitness measure in case of the PSDC is lower than both the K-means and the Modified Ant technique. As illustrated before, the lower values of the fitness measure indicates better performance.

TABLE 2
COMPARING DUNN INDEX'S VALUES FOR PSDC TECHNIQUE, K-MEANS AND MODIFIED ANT CLUSTERING ALGORITHM, FOR DATA SETS OF DEFFIRENT SIZES AND DIFFERENT NUMBER OF CLUSTERS.

Dunn index							
		PSDC		Kmeans		Ant	
N	φ	itr	Index value	itr	Index value	itr	Index value
1000	5	2	1.4699	6	1.0348	10	1.444
1800	9	2	1.461	12	0.176	10	1.101
2800	7	2	1.4752	12	0.1698	10	1.0123
3200	8	2	1.4762	10	0.1671	10	1.1947
3600	9	2	1.4666	14	0.1616	10	1.1589

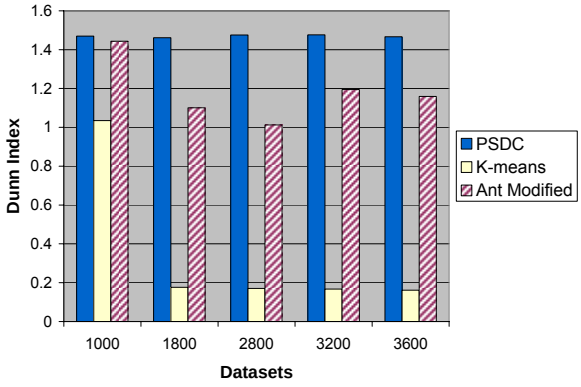

Fig. 4. Comparing Dunn index results for PSDC technique, K-means and Modified Ant algorithm for experimental datasets of different sizes and distributions.

As illustrated from table 2 and Fig. 4 we can observe that the Dunn Index remains high for the PSDC algorithm applied on all datasets of different sizes. As the data size increases, the improvement of the Dunn Index indicator is extremely higher than both the K-means and the Modified Ant technique.

TABLE 3
COMPARING DAVIES-BOULDEN INDEX'S VALUES FOR PSDC TECHNIQUE, K-MEANS AND MODIFIED ANT CLUSTERING ALGORITHM, FOR DATA SETS OF DEFFIRENT SIZES AND DIFFERENT NUMBER OF CLUSTERS.

Davies-Boulden							
		PSDC		Kmeans		Ant	
N	φ	itr	Index value	itr	Index value	itr	Index value
1000	5	2	0.412	6	0.5553	10	0.4146
1800	9	2	0.4113	12	0.7541	10	0.5292
2800	7	2	0.4066	12	0.7596	10	0.6431
3200	8	2	0.4071	10	0.7403	10	0.5674
3600	9	2	0.4068	14	0.7003	10	0.5819

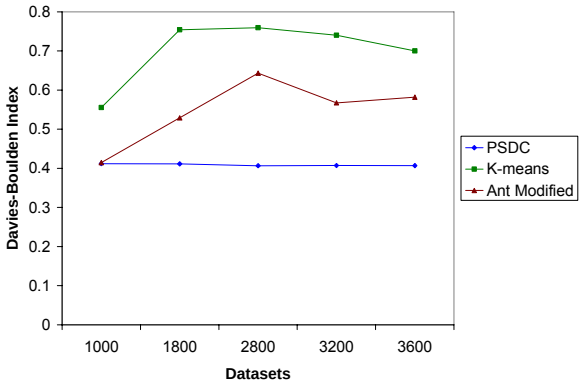

Fig. 5. Comparing Davies-Boulden index results for for PSDC technique, K-means and Modified Ant algorithm for experimental datasets of different sizes and distributions.

As observed from table 3 and Fig. 5, the cluster solutions returned by the PSDC have showed better Davies-Boulden index over both the Modified Ant technique and the K-means algorithm. As illustrated in the previous sections, lower values for the Davies-Boulden index indicates better performance.

TABLE 4
COMPARING INNER CLUSTER VARIANCE'S VALUES FOR PSDC TECHNIQUE, K-MEANS AND MODIFIED ANT CLUSTERING ALGORITHM, FOR DATA SETS OF DEFFIRENT SIZES AND DIFFERENT NUMBER OF CLUSTERS.

Inner cluster variance							
		PSDC		Kmeans		Ant	
N	φ	itr	Index value	itr	Index value	itr	Index value
1000	5	2	333.041	6	448.4443	10	336.431
1800	9	2	593.656	12	9.94E+02	10	648.782
2800	7	2	459.152	12	908.7493	10	506.899
3200	8	2	524.250	10	9.16E+02	10	569.200
3600	9	2	549.000	14	937.0185	10	594.00

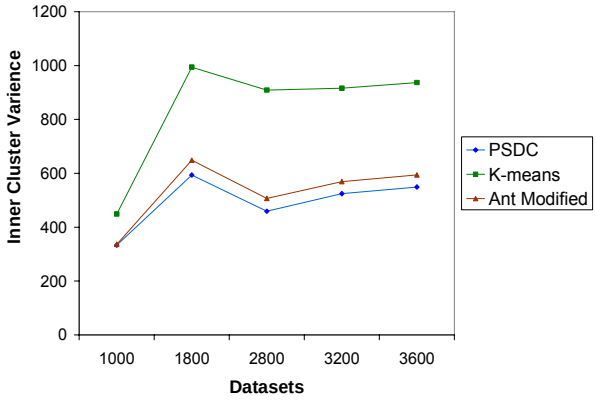

Fig. 6. Comparing inner cluster variance results for PSDC technique, K-means and Modified Ant for experimental datasets of different sizes and distributions.

As shown in Fig. 6 and table 4, we can observe how much the PSDC illustrates improvement in the inner cluster variance over both the Modified Ant technique and the K-means clustering. As illustrated in the previous sections, lower values for the Inner cluster variance index indicates better performance.

TABLE 5
COMPARING SD VALIDITY INDEX'S VALUES FOR PSDC TECHNIQUE, K-MEANS AND MODIFIED ANT CLUSTERING ALGORITHM, FOR DATA SETS OF DEFFIRENT SIZES AND DIFFERENT NUMBER OF CLUSTERS.

SD validity							
		PSDC		Kmeans		Ant	
N	φ	itr	Index value	itr	Index value	itr	Index value
1000	5	2	0.0688	6	0.1184	10	0.0866
1800	9	2	0.0672	12	0.221	10	0.11332
2800	7	2	0.0687	12	0.2063	10	0.15212
3200	8	2	0.0684	10	0.2317	10	0.157067
3600	9	2	0.067	14	0.2284	10	0.1786

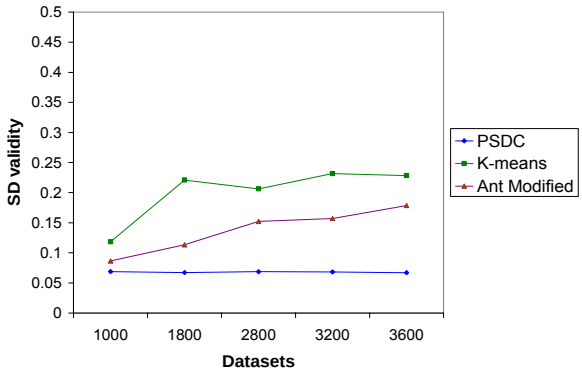

Fig. 7. Comparing SD validity index results for PSDC technique, K-means and Modified Ant algorithm for experimental datasets of different sizes and distributions.

As illustrated in table 5 and Fig. 7, we can observe better results for the PSDC over both the Modified Ant clustering algorithm and the K-means algorithm. As the data size increases, the improvement in the SD validity index is

considerably large. As illustrated in the previous sections, lower values for the SD validity index indicates better performance.

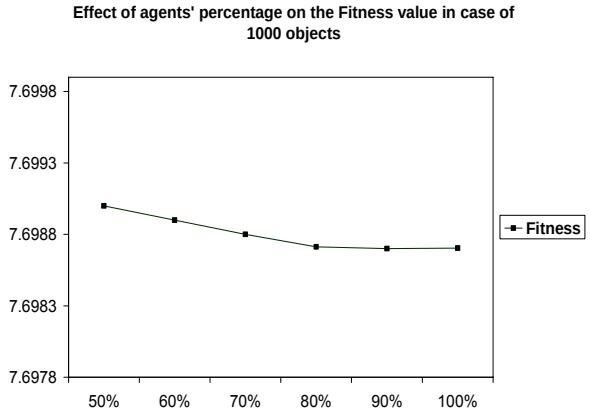

Fig. 8. Studying the effect of agent's percentage on the fitness value in case of 1000 data objects.

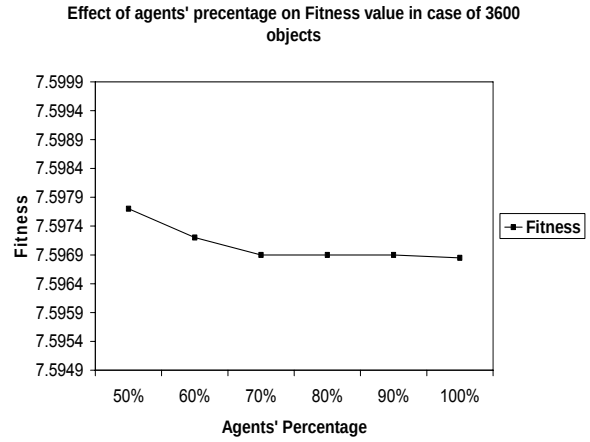

Fig. 9. Studying the effect of agent's percentage on the fitness value in case of 3600 data objects.

Fig. 8 and Fig. 9 show the effect of the generated number of agents (population size) on the fitness measure in case of 1000 data objects and 3600 data objects, respectively. As shown in the figures, the fitness measure improves as the percentage of the generated number of agents per population increases up to an asymptotic point above which no further improvement is recognized. Setting the number of agents to 80% of the data size seems to be very reasonable. Any further increase in the generation size will not improve the results but may results in further delayed solutions.

Fig. 10 illustrates an example of the clustering process using PSDC in 2D dimension & in geometric form. In fig 10a the data objects are generated while in fig 10b the agents were assigned randomly to the data objects with still no movement. In fig 10c the first iteration took place and the agents' rules were applied and agents moved as shown. At the end of the iterations the agents decided that there are 3 clusters according to the guiding rules and reached the centers of the approximated clusters. Then, performance indicators will be calculated for the current cluster solution.

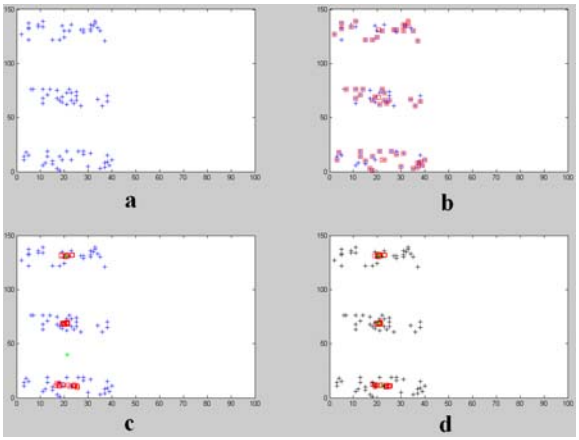

Fig. 10. Three clusters of synthetic data in geometric form with agents discovering each cluster.

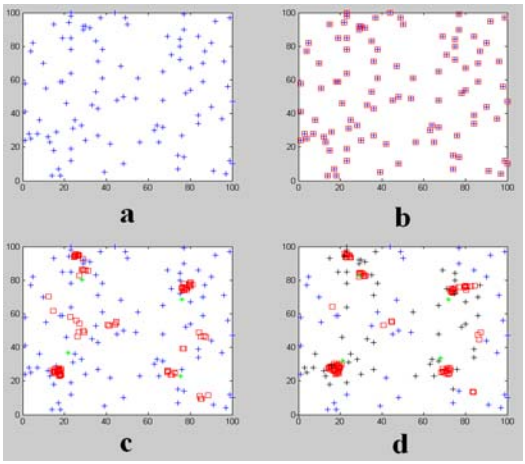

Fig. 11. Four clusters of synthetic data in distributed form with agents discovering each cluster

Fig. 11 illustrates an example the clustering process using PSDC in 2D dimension & distributed form In fig. 11a we observe the data objects before introducing the agents then in fig. 11b the agents are assigned randomly to the data objects. Fig 11c shows the situation after the first iteration and after applying the agents' rule. Finally in fig. 11d we got the agents after the final iteration in which they decided that there are 4 clusters according to the explained rules and found also each center and ready for indicators' calculations.

C. Using real data sets

Three real data collections taken from the Machine Learning Repository to compare our results. The Iris data contains 150 items described by 4 attributes. Its 3 clusters are each of size 50. Two of them are linearly non-separable. The yeast data contains 1484 data elements described by 8 attributes. The Breast Cancer Wisconsin data contains 699 items described by 9 attributes. Its 2 clusters are of size 458 and 241 respectively. The following table shows a clear comparison of well known clustering algorithms with the proposed algorithm, applied on real data sets.

TABLE 6

COMPARING THE PERFORMANCE OF THE PSDC ALGORITHM WITH HIERARCHICAL AGGLOMERATIVE AVERAGE-LINK CLUSTERING, THE ONE-DIMENSIONAL SELF-ORGANIZING MAPS, AND THE K-MEANS, APPLIED ON REAL DATA SETS.

Iris	**k-means**	**average link**	**1D-SOM**	**PSDC**
No. of clusters	3	3	3	**3**
Dunn Index	**2.65**	2.51	2.08	2.51
Variance	0.92	0.9	**0.89**	**0. 9**
Runtime	0.16	**0.02**	0.08	2. 3
Yeast	**k-means**	**average link**	**1D-SOM**	**PSDC**
No. of clusters	10	10	10	**10**
Dunn Index	1.7	1.56	1.22	**1.8**
Variance	**1.54**	1.6	1.7	1.57
Runtime	**1.7**	14	12.16	13
Breast Cancer WISCONSIN	**k-means**	**average link**	**1D-SOM**	**PSDC**
No. of clusters	2	2	2	2
Dunn Index	**5.47**	4.92	5.46	**5.46**
Variance	**1.61**	1.63	1.61	**1.62**
Runtime	0.40	1.65	2.5	11.3

Table (6) shows the results for K-means, hierarchical agglomerative average-link clustering, one-dimensional self-organising maps and the proposed Particle Swarm-Based Data Clustering (PSDC), applied on three real datasets. The quality of the clustering is measured using the Dunn Index, Inner Cluster Variance and runtimes. The bold face indicates the best value. The two inseparable clusters in the iris data affected the results as shown and reflected in the performance indicators values and the runtime. But the PSDC identified the close second best result in both the Dunn Index and the Inner Cluster Variance. In the yeast data the clusters are identified successfully with better value in the Dunn Index and second best result in the Inner Cluster Variance. We can notice the runtime value is the main drawback of the PSDC algorithm in case of well separated clusters but results in a better performance over other algorithms.

VI. CONCLUSION

In this paper a new particle swarm-based data clustering technique had been introduced. The mechanism showed outstanding performance compared to other methods for clustering. Nonetheless, the mapping simultaneously provided by the algorithm is one of the most attractive features. The algorithm has a number of features that make it an interesting candidate for cluster analysis. First, its linear scaling behaviour, which make it suitable for use on large data sets. Also, the nature of the algorithm makes it fairly robust to the effects of outliers within the data. In addition, swarm-based clustering has the capacity to work with any kind of data that can be described in terms of symmetric dissimilarities, and it imposes no assumptions on the shape of the clusters it works with. Finally, an important strength of the algorithm is its ability to automatically determine the number of clusters within the data. The adaptation scheme proposed in the algorithm make it possible to tune with structures exist within the data. In addition, the guiding rules, which are alignment, cohesion and separation rule, led to better solutions. In each iteration, each single agent updates its position and its population could pass useful information to the next one resulting in reaching better clustering solution faster and in less time. These new features allow the algorithm to be highly dynamic. The proposed PSDC algorithm illustrated superior performance when compared with both the modified ant clustering algorithm and the k-means algorithm. Moreover, the performance of the PSDC showed comparative performance when compared with the hierarchical agglomerative average-link clustering and the one-dimensional self-organising maps applied on real data sets. Our approach performs favourably selecting good partitioning decisions, and avoids being trapped in local optimal solutions by making use of the cooperative and adaptive behaviour of the agents.

REFERENCES

[1] M. G. H. Omran, A. P. Engelbrecht, and Ayed Salman, "Dynamic Clustering using Particle Swarm Optimization with Application in Unsupervised Image Classification", *Transactions on engineering., computing and technology,* vol. 9, pp.199-204, 2005.

[2] A.K. Jain, M. N. Murty, and P.J. Flynn, "Data clustering: A review, *ACM Computing Surveys*, vol. 31, no. 3, pp.264–323, 1999.

[3] X. Cui, J. Gao and T. E. Potok, "A Flocking Based Algorithm for Document Clustering Analysis", *Journal of System Architecture*, Elsevier Science Inc, vol.52, pp.505-515, 2006.

[4] H. Van Dyke Parunak, R. Rohwer, T. C. Belding, S. Brueckner, "Dynamic Decentralized Any-Time Hierarchical Clustering", *the 29th Annual Int. ACM SIGIR Conference on Research & Development on Information Retrieval*, Seattle, USA, pp. 66-81, 2006.

[5] X. Cui, P. Palathingal, T.E. Potok, "Document Clustering using Particle Swarm Optimization", *IEEE Swarm Intelligence Symposium*, Pasadena, California, pp.185-191, 2005.

[6] S. Bandyopadhyay, U. Maulik, "An evolutionary technique based on K-Means algorithm for optimal clustering in R^N", *Information Sciences, Elsevier Science Inc.,* New York, NY, USA, pp.221-237, 2002.

[7] A. L. fred, "Finding Consistent Clusters in Data Partitions", *Lecture Notes In Computer Science*, vol. 2096, Springer-Verlag, London, UK, pp.309-318, 2001.

[8] G. Jones, A. Robertson, C. Santimetvirul, P. Willett, Nonhierarchic, "Document clustering using a genetic algorithm", *Information Research*, vol. 1, no. 1, 1995.

[9] J. Vesanto, E. Alhoniemi, "Clustering of the self-organizing map", *IEEE Transactions on Neural Networks*, vol. 11, no. 3, pp.586–600, 2000.

[10] C. Aranha, H. Iba, "The effect of using evolutionary algorithms on ant clustering techniques", *the 3rd Asian-*

Pacific workshop on Genetic Programming (ASPGP06), pp.24-34, 2006.

[11] J. Handl, J. Knowles, M. Dorigo, "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, 2003.

[12] C.F. Tsai, H.C. Wu, C.W. Tsai, "A new clustering approach for data mining in large databases", *the International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN'2002)*, *IEEE Computer Society*, pp.1087-1089, 2002.

[13] X.B. Yang, J.G. Sun, D. Huang, "A new clustering method based on ant colony algorithm", *the 4th world congress on intelligent control and automation*, pp. 2222-2226, June 2002.

[14] F. Kovács, C. Legány, A. Babos, "Cluster Validity Measurement Techniques", *the 6th International Symposium of Hungarian Researchers on Computational Intelligence*, Budapest, 2005.

[15] N. Bolshakova, F. Azuaje. "Improving expression data mining through cluster validation", *the 4th Annual IEEE EMBS Special Topic Conference on Information Technology Applications in Biomedicine*, pp.19-22, 2003.

[16] B. Stein, S. Meyer zu Eissen, F. Wißbrock, "On Cluster Validity and the Information Need of Users*", the 3rd IASTED Int. Conference on Artificial Intelligence and Applications (AIA 03)*, Benalmádena, Spain, ACTA Press, pp.216-221, September 2003.

[17] N. Bolshakova, F. Azuaje, "Cluster validation techniques for genome expression data classification", *SignalProcessing*, vol.83, no.4, pp.825-833, 2003.

[18] C. Blake, C. Merz. UCI repository of machine learning databases. Technical report, Department of Information and Computer Sciences, University of California, Irvine, 1998. Available at: Http:://www.ics.uci.edu/_ mlearn/MLRepository.html.

[19] J. Handl, M. Dorigo, "On The Performance of Ant-based Clustering *", the 3rd Int. Conf. on Hybrid Intelligent Systems*, IOS Press, Australia, 2003.