Preprint submitted to Applied Soft Computing

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Finding groups in data: Cluster analysis with ants

Urszula Boryczka *, Marcin Budka

Institute of Computer Science, University of Silesia, Bedzinska 39, 41-200 Sosnowiec, Poland

School of DEC, Bournemouth University, Fern Barrow, BH12 5BB, Poole UK

ARTICLE INFO

Article history:

Keywords:
Ant-based clustering algorithm
Data clustering
Visual data clustering
Classification

ABSTRACT

We present in this paper a modification of Lumer and Faieta's algorithm for data clustering. This approach mimics the clustering behavior observed in real ant colonies. This algorithm discovers automatically clusters in numerical data without prior knowledge of possible number of clusters. In this paper we focus on ant-based clustering algorithms, a particular kind of a swarm intelligent system, and on the effects on the final clustering by using during the classification different metrics of dissimilarity: Euclidean, Cosine, and Gower measures. Clustering with swarm-based algorithms is emerging as an alternative to more conventional clustering methods, such as e.g. *k*-means, etc. Among the many bio-inspired techniques, ant clustering algorithms have received special attention, especially because they still require much investigation to improve performance, stability and other key features that would make such algorithms mature tools for data mining.

As a case study, this paper focus on the behavior of clustering procedures in those new approaches. The proposed algorithm and its modifications are evaluated in a number of well-known benchmark datasets. Empirical results clearly show that ant-based clustering algorithms performs well when compared to another techniques.

1. Introduction

The collective behaviors and self-organization of social insects have inspired computer scientists to perform computer simulations to replicate this behavior. There are two main reasons: firstly, these mechanisms responsible for the behaviors are yet unknown and therefore we can better understand the nature. The second reason is that the behavior of social insects has many attractive features such as robustness and reliability. Computer models of these behaviors, based on the clustering and sorting of insects can lead to better performance in areas such as search, data mining, and experimental data analysis.

In the last two decades, many advances in algorithmica have been based on the observation of the natural world. Biomimicry—applications of swarm intelligence have been developed especially in the optimization field. The swarm intelligent systems are quite easy to adapt, and knowledge of individual behaviors and interactions is not very complicated. Rather, these behaviors and interactions emerge from very simple rules. Bonabeau et al. [3] define swarm intelligence as "the emergent collective intelligence of groups of simple agents". We agree with the core of this definition and we want to emphasize the emergent behavior (self-organization), simple processes leading to complex results. In the words of one mathematician, Stephen Wolfram: "It is possible to make things of great complexity out of things that are very simple. There is no conservation of simplicity" [25].

Self-organization in social insects is interpreted through four main mechanisms:

(1) The existence of multiple interactions.
(2) Application through positive feedback.
(3) Negative feedback.
(4) Application of fluctuations.

Ants foraging process in some species has been analyzed by Deneubourg et al. [9]. He notably showed how ants can find the best (shortest) way to reach a resource. In a nutshell, the accumulation of pheromones is faster on the shortest route, so positive feedback therefore gives it priority. On this basis Dorigo and Stützle [10] proposed the concept of Ant Colony Optimization. Dorigo and many other researchers applied this mechanism to many combinatorial optimization problems such as TSP, JSP and then extended it to a whole class of optimization problems. Such algorithms can now be found in telecommunications routing, to design of electronic circuits or – for example – the organization of industrial processes.

Biomimicry of social insects focuses into observing how nature solves situations that are similar to different optimization problems we face. The study of ant colonies has offered remarkable insight in this field—not only in the combinatorial optimization but also ant colonies can provide new ideas for clustering techniques.

* Tel.: +48 32 291 82 83; fax: +48 32 291 82 83.
E-mail address: urszula.boryczka@us.edu.pl.

Among the social insects' behaviors, the most widely recognized is the ants' ability to work as a group in order to finish a task that cannot be finished by a single agent. Also seen in human society, this ability of ants is a result of cooperative effects. The cooperative effect refers to the phenomenon that the effect of two or more individuals or parts coordinating is higher than the total of the individual effects. Some researchers have achieved promising results in data mining by using the artificial ant colony. The high number of individuals in ant colonies and the decentralized approach to task coordination means that ant colonies show high degrees of parallelism, self-organization and fault tolerance. These features are desired characteristics in modern optimization techniques.

In this paper, a novel ant-based clustering algorithm is proposed to improve the performance of many *k*-medoids-based algorithms. A new version of ant-based clustering algorithm ACA is inspired from the behavior of real ants. The paper is organized as follows: Section 2 gives a detailed description of the different approaches to clustering. Section 3 presents methodology of clustering by ants. Section 4 describes a biological inspirations in clustering algorithms. In the next section an ant-based clustering algorithm and its modifications is presented. Section 6 presents the experiments that have been conducted to see the influence of modifications and statistic measures regardless on different datasets. Results of the experimental part of this article; validations of those approaches are shown in Section 7. The last section concludes and discusses future evolutions of ant-based clustering algorithms.

2. Different approaches to clustering

Clustering problems have been discussed extensively in the database literature as a tool for similarity search, customer segmentation, pattern recognition, trend analysis and classification. Various methods have been studied in considerable detail by both the statistics and database communities [1,4,15,29]. Detailed survey on clustering methods can be found in Refs. [11,22,24,26,34].

Clustering is a form of classification imposed over a finite set of objects. The goal of clustering is to group sets of objects into classes such that similar objects are placed in the same cluster while dissimilar objects are in separate clusters. Clustering (or classification) is a common form of data mining and has been applied in many fields including data compression, texture segmentation, vector quantization, computer vision and various business applications. Some algorithms assume that the number of clusters is prespecified as a user parameter. Various objective functions may be used in order to make a quantitative determination as to how well the points are clustered.

The essential part of clustering is to classify all objects into several groups so as to achieve some optimal conditions. Most conventional clustering methods would rapidly become computationally interactable as the problem scale gets larger due to the combinatorial nature of the methods. Brucker [5] and Welch [33] proved that, for specific objective functions clustering becomes an NP-hard problem when the number of clusters exceeds three. Hansen and Jaumard [21] pointed out that even though the best algorithms developed for some specific objective functions, there would exhibit complexities of $O(N^3 \log N)$ or $O(N^3)$, so further improvements can fulfill this gap.

Five categories of heuristic algorithms for clustering were determined (according forms of heuristics used in these approaches):

- statistics clustering;
- mathematical programming;
- network programming;
- neural network;
- metaheuristics.

The algorithms for conventional statistic clustering include agglomerative hierarchical clustering method, divisive hierarchical clustering method, *k*-means, etc. The algorithms for mathematical programming range from dynamic programming, Lagrangian relaxation, linear relaxation, column generation, branch-and-price and Lipschitz continuous. The algorithms for neural network mainly include self-organizing map (SOM) and adaptive resonance theory. The algorithms for metaheuristics are rapidly developed recently, including evolutionary algorithms, Tabu Search, Simulated Annealing and Ant Colony Optimization. These algorithms have also been validated by comparing with hybrid methods, fast self-organizing map combining with *k*-means and genetic *k*-means approach and many others. There exists a large number of clustering algorithms in the literature including *k*-means [28], *k*-medoids [24], CACTUS [14], CURE [16], CHAMELEON [23] and DBSCAN [12]. No single algorithms is suitable for all types of objects, nor all algorithms appropriate for all problems, however, the *k*-medoids algorithms have been shown to be robust to outliers [24], compared with centroid-based clustering. Partitioning Around Medoids (PAM) [24], Clustering LARge Applications (CLARA) [24] and Clustering Large Applications based on RANdomized Search (CLARANS) [29] are three popular *k*-medoids-based algorithms while the Clustering Large Applications based on Simulated Annealing (CLASA) algorithm applies simulated annealing to select better medoids [7]. The drawback of the *k*-medoids algorithms is the time complexity of determining the medoids.

3. Methodology of clustering by ants

The process of cluster analysis consists of three major stages: feature extraction, similarity computation and grouping. In this first phase we establish the main features of objects and the method of comparison. The next stage shows the similarity between the objects take into consideration in term of these chosen features, attributes. The result of similarity or dissimilarity computation is presented in the next step—grouping, the form of partitioning these objects into groups. Ant clustering method involves only two last steps of the process of clustering.

The major difference observed in ant clustering algorithms and another clustering systems is that ants can analyze the data on toroidal bi-dimensional grid, which cannot show directly information about disparity between two different pieces of data as it happens in *n*-dimensional space (where *n* determine the dimensionality of the data). The swarm of ants reside in an environment consisting of objects that may be picked up or dropped in appropriate position. A grid in the environment may contain one ant, one object or both one ant and one object. The environment—workspace of ants consists of two elements. The first is a collection of objects that in the beginning are randomly dispersed throughout the workspace, and as time goes by are moved by ants using special rules. The second component of this workspace is a swarm of ants which can move around and pick up and drop the objects. All moves occur in discrete time steps. An important characteristic of the environment is the relationship between the size of the environment, the number of objects, and the number of ants.

If an ant is not currently carrying an object it may attempt to pick it in a moment when it is located in the same grid in the workspace as the ant itself. The probability of picking or dropping an object depends on the distance in feature space between that object and other objects in its neighborhood. At each time step, after decision making, the ant performs a random movement on the workspace.

Objects that are near each other in the workspace will be likely to be dropped in neighboring positions. After the initial phase, a small cluster of few similar objects will form. During the formation of clusters we observe a stigmergetic process, so the probability of dropping new, similar objects near it is greater than anywhere else on the workspace. This leads to a process of a positive feedback which produces a greater number of objects in the analyzed clusters.

4. Biological inspirations and algorithms

Clustering and sorting behavior of ants has stimulated researches to design new algorithms for data analysis and partitioning. Several species of ants cluster corpes to form a "cemetery", or sort their larvae into several piles. This behavior is still not fully understood, but a simple model, in which ants move randomly in space and pick up and deposit items on the basis of local information, may account for some of the characteristic features of clustering and sorting in ants [3].

In several species of ants, workers have been reported to form piles of corpes – cemeteries – to clean the nests. Chretien [6] has performed experiments with the ant *Lasius niger* to study the organization of cemeteries. Other experiments on the ant *Phaidole pallidula* are also reported in Ref. [9]. Brood sorting is observed in the ant *Leptothorax unifasciatus* [13]. Workers of this species gather the larvae according to their size. Franks and Sendova-Franks [13] have intensively analyzed the distribution of brood within the brood cluster (Fig. 1).

Deneubourg et al. [9] has proposed two closely related models to account for the two above-mentioned phenomena of corpse clustering and larval sorting in ants. As we mentioned above, general idea is that isolated items should be picked up and dropped at some other location where more items of that type are present. In this way, the system proposed by Deneubourg was able to realize clustering in a global scale. Let us assume that there is only one type of item in the environment. The probability p_p for a randomly moving unladen agent to pick up an item is given by

$$p_p = \left(\frac{k_1}{k_1 + f}\right)^2$$

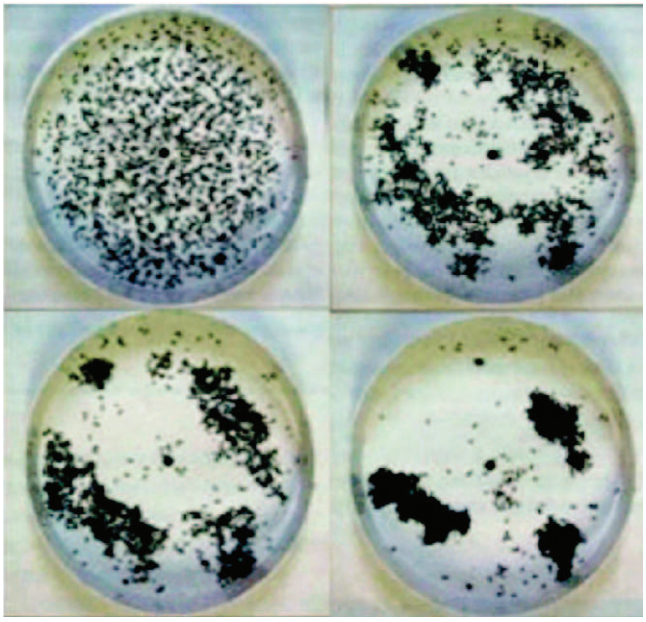

Fig. 1. Real ants cluster [3].

where

- f is the perceived fraction of items in the neighborhood of the agent,
- k_1 is a threshold item.

The probability p_d for a randomly moving loaded agent to deposit an item is given by

$$p_d = \left(\frac{f}{k_2 + f}\right)^2$$

where k_2 is another threshold constant.

Deneubourg et al. [8] have assumed that f is computed through a short-term memory that each agent possesses, it is simply the number N of items encountered during these last T time units, divided by the largest possible number of items that can be encountered during this time.

Gutowitz [17] has suggested the use of spatial entropy to track the dynamics of clustering. The entropy level of work area was determined by the presence or absence of objects, so that a place completely full or empty would have the lowest entropy, and a checkered pattern would have the highest. The level of entropy of their surroundings would provoke the ants to take an action. In this way, in areas with low entropy the ants would not try to pick or drop items. These complexity-seeking ants were thus able to avoid actions that did not contribute to the clustering process, performing their actions more efficiently. The spatial entropy E_s at scale s is defined by

$$E_s = \sum_{I \in S} P_I \log P_I$$

where P_I is the fraction of all objects on the lattice that are found in s-patch I.

Oprisan et al. [30] proposed a variant of Deneubourg basic model (hereafter called BM), in which the influence of previously encountered objects is distributed by a time factor.

Bonabeau [2] also explored the influence of various weighting functions, especially those with short-term activation and long-term inhibition.

Lumer and Faieta [27] have generalized Deneubourg et al.'s BM to apply it to exploratory data analysis. The idea is to define a distance or dissimilarity d between objects in the space of object attributes:

- if two objects are identical then $d(o_i, o_j) = 0$,
- when two objects are not identical then $d(o_i, o_j) = 1$.

The algorithm introduced by Lumer and Faieta (hereafter LF) consists of projecting the space of attributes onto some lower dimensional space, typically of dimension $z = 2$. Let us assume that an ant is located at side r at time t, and finds an object o_i at that site. The "local density" $f(o_i)$ with respect to object o_i is given by

$$f(o_i) = \begin{cases} \dfrac{1}{s^2} \displaystyle\sum_{o_j \in \text{Neigh}(s\times s)(r)} \left[1 - \frac{d(o_i, o_j)}{\alpha}\right], & \text{when } f > 0 \\ 0, & \text{otherwise} \end{cases}$$

where

- $f(o_i)$ is a measure of the average similarity of object o_i with the other objects o_j present in the neighborhood of o_i,
- α is a factor that defines the scale for dissimilarity: it is important for it determines when two items should or should not be located next to each other.

Lumer and Faieta [27] define picking up and dropping probabilities as follows:

$$p_{\mathrm{p}}(o_i) = \left(\frac{k_1}{k_1 + f(o_i)}\right)^2$$

$$p_{\mathrm{d}}(o_i) = \begin{cases} 2\,f(o_i), & \text{when } f(o_i) < k_2 \\ 1, & \text{when } f(o_i) \geq k_2 \end{cases} \quad (1)$$

where k_1, k_2 are two constants that play a role similar to k_1 and k_2 in the BM.

High-level description of the Lumer–Faieta algorithm is presented below:

LF algorithm

```
0  /*initialization*/
1  for every object o_i do
2    Place o_i randomly on grid
3  end for
4  for all ants do
5    place ant at randomly selected site
6  end for
7  /*main loop*/
8  for all ants do
9    for t = 1 to t_max do
10     if ((agent unladen) and (site occupied by item o_i)) then
11       Compute F(o_i) and p_p(o_i)
12       Draw random real number R ∈ (0,1)
13       if (R ≤ p_p(o_i)) then
14         Pick up item o_i
15       end if
16     else
17       if (agent carrying item o_i) and (site empty)) then
18         Compute f(o_i) and p_d(o_i)
19         Draw random real number R ∈ (0,1)
20         if (R ≤ p_d(o_i)) then
21           Drop item
22         end if
23       end if
24     end if
25     Move to randomly selected neighboring site not occupied
26      by other agent
27   end for
28 end for
29 Print location of items.
```

5. Ant-based clustering algorithms—ACA and ACAM

The ant-based clustering algorithms are mainly based on versions proposed by Deneubourg, Lumer and Faieta. A number of slight modifications have been introduced that improve the quality of the clustering and, in particular, the spatial separation between clusters on the grid. Recently Handl and Meyer [20] extended Lumer and Faieta's algorithm and proposed an application to the classification of Web documents. The model proposed by Handl and Meyer has inspired us to use this idea to classical cluster analysis. The basic idea is to pick up or drop a data item on the grid.

5.1. Classical approach—ACA

We also have employed a modified version of the "short-term memory" introduced by Lumer and Faieta in Ref. [27]. Each ant has a permition to exploit its memory according these rules: if an ant situated at grid cell p, and carrying a data item i, it uses its memory to proceed to all remembered positions, one after the other. Each of them is evaluated using the neighborhood function $f^*(i)$ for finding a dropping site for the currently carried data item i.

For picking and dropping decisions the following threshold formulae are used:

$$p^*_{\mathrm{pick}}(i) = \begin{cases} 1, & \text{if } f^*(i) > 1 \\ \dfrac{1}{f^*(i)^2}, & \text{else} \end{cases},$$

$$p^*_{\mathrm{drop}}(i) = \begin{cases} 1, & \text{if } f^*(i) \geq 1 \\ \dfrac{1}{f^*(i)^4}, & \text{else} \end{cases},$$

where $f^*(i)$ is a modified version of Lumer and Faieta's neighborhood function:

- $$f^*(i) = \begin{cases} \frac{1}{\sigma^2}\sum_j \left[1 - \frac{d(i,j)}{\alpha}\right], & \text{if } f^* > 0, \text{and } \left(1 - \frac{d(i,j)}{\alpha}\right) > 0 \\ 0, & \text{otherwise,} \end{cases}$$
- $1/\sigma^2$ is a neighborhood scaling parameter,
- α is a parameter scaling the dissimilarities within the neighborhood function $f^*(i)$,
- $d(i,j)$ is a dissimilarity function.

Ant-based clustering algorithm requires a number of different parameters to be set, which have been experimentally observed. Parameters of this algorithm we can divide into two groups:

(1) To be independent of the data.
(2) To be set as a function of the size of the dataset.

The first group includes:

- the number of agents, which is set to be 10,
- the size of the agents' short-term memory, which we equally set to 10,
- the initial clustering phase (from t_{start} to t_{end}): $t_{\text{start}} = 0.45N$, $t_{\text{end}} = 0.55N$, where N denote the number of iterations,
- we replace the scaling parameter $1/\sigma^2$ by $1/N_{\text{occ}}$ after an initial clustering phase, where N_{occ} is the actual observed number of occupied grid cells within the local neighborhood.

The employed distance function is the Euclidean measure for the initial testing and the Cosine and Gower measures for the next step of the data analysis.

Several parameters should be selected in dependence of the size of the dataset tackled. Given a set of $N_{\max}$ items, the grid should offer a sufficient amount of "free" space to permit the quick dropping of data items. This can be achieved by

- using a square grid with resolution of $\sqrt{10N_{\max}} \times \sqrt{10N_{\max}}$,
- the step should permit sampling of each possible grid position within one move, which is obtained by setting it to stepsize: $\sqrt{20N_{\max}}$,
- the number of iterations: $\sqrt{2000N_{\max}}$, with a minimal number of 1,000,000.

During the sorting process, α determines the percentage of data items on the grid that are classified as similar, such that: a too small choice of α prevents the formation of clusters on the grid; on the other hand, a too large choice of α results in the fusion of individual clusters, and in the limit, all data items would be gathered within one cluster.

The scheme for α-adaptation used in this application is a part of a self-adaptation of agents activity. A heterogenous population of ants is used in the standard ant-based clustering algorithm (ACLA)—with its own parameter α. An agent considers an adaptation of its own parameter after it has performed N_{active} moves. During this time, it keeps track of the failed dropping operations N_{fail}. The rate of failure is determined as $r_{\text{fail}} = N_{\text{fail}}/N_{\text{active}}$ where N_{active} is fixed to 100. The agent's parameter α is then updating using the rule:

$$\alpha = \begin{cases} \alpha + 0.01, & \text{if } r_{\text{fail}} > 0.99 \\ \alpha - 0.01, & \text{if } r_{\text{fail}} \le 0.99. \end{cases}$$

5.2. Modifications of ACA—ACAM

For increasing the robustness of ant-based clustering we also examine some improvements. The modified version of ACAM has incorporated two main modifications in relation to ACA:

- an adaptive perception scheme occurred in the density function,
- a cooling scheme of α-adaptation.

The neighborhood function or density function $f^*(i)$ depends on the perception field s^2, of each ant. The stable value of parameter s may sometimes cause inadequate behaviors, because it is not possible to distinguish the differences between clusters of different sizes. On the other hand, a large perception field may be useful at the beginning of our algorithm, when data are scattered at random manner on the grid file.

In order to overcome this difficulty a new proposition of the density function with a new scalable parameter s_0^2/s^2 (of the relative perception field coefficient) is proposed:

$$f^*(i) = \begin{cases} \frac{s_0^2}{s^2}\sum_j \left[1 - \frac{d(i,j)}{\alpha}\right], & \forall o_j \left(1 - \frac{d(i,j)}{\alpha}\right) > 0 \\ 0, & \text{otherwise} \end{cases}$$

where

- s_0^2/s^2 is a new neighborhood scaling parameter, a relationship between the initial and current size of perception,
- α is a parameter scaling the dissimilarities within the neighborhood function $f^*(i)$,
- $d(i,j)$ is a dissimilarity function.

The next modification is strongly connected with the parameter α and concerns the methods of its changes. In a nutshell, a cooling scheme is adopted in our proposed algorithm. This scheme is really simple: after η_{update} iterations has passed, the value of the parameter α starts being increased if a random value r is smaller than $p(\Delta f_{\text{avg}})$. This new α-adaptation scheme that we propose in our approach is computed as follows:

$$\alpha = \begin{cases} \alpha - 0.01, & \text{if } \Delta f_{\text{avg}} \le 0 \\ \alpha + 0.01, & \text{if } r < p(\Delta f_{\text{avg}}), \end{cases}$$

where

- r is the random number $r \in [0,1]$,
- $f(o_i) = (1/n)\sum_{o_j \in \text{Neigh}(3\times 3)(r)} d(o_i, o_j)$,
- n is the number of objects in the nearest neighborhood of object o_i.

Value of $p(\Delta f_{\text{avg}})$ is determined as follows:

$$p(\Delta f_{\text{avg}}) = \mathrm{e}^{-\Delta f_{\text{avg}}/T},$$

similarly to the acceptance criterion in Simulated Annealing, where

- $\Delta f_{\text{avg}} = f_{\text{avg}} - f'_{\text{avg}}$, the difference between previous and current (after η_{update} number of iterations) values of f,
- $f_{\text{avg}} = (1/N)\sum_i^N f^*(o_i)$,
- N is the number of classified objects,
- $T \le 0.03$ is a parameter of the cooling procedure.

High-level description of the ant clustering algorithm (ACA) is presented below:

ACA algorithm

```
/*Initialization Phase*/
Randomly scatter objects on the grid file
for each agent a_j do
   random_select_object (o_i)
  pick_up_object (o_i)
   place_agent a_j at randomly selected empty grid location
end for
{*Main loop*}
  for t = 1 to t_max do
    random_select_agent (a_j)
    move_agent a_j to new location
    i = carried_object(agent a_j)
    Compute f*(o_i) and p*_drop(o_i)
    if drop = True then
      while pick = False do
        i = random_select_object (o_i)
        Compute f*(o_i) and p*_pick(o_i)
        Pick_up_object o_i
      end while
    end if
  end for
end
```

By doing so, more suitable and strongly correlated objects can be clustered, in these way the ACAM will also tend to converge to better solutions.

6. Experimental results

In order to evaluate the resulting partition obtained by ACLA we have set up the following method. The first datasets used to illustrate the performance of the algorithms was a modified version of the well-known datasets proposed to study the standard ant-based clustering algorithm [19]. The square datasets are the most popularly used type of datasets. They are two-dimensional and consists of four clusters arranged as a square. To conform to distributed datasets the data are spread uniformly among the various sites.

Secondly, we have applied ant-based clustering algorithms to real world databases from the Machine Learning repository which are often used as a benchmark. The dataset is useful to show experimentally the efficiency of ACA on data with known properties end difficulty. The real data collections used were the Iris data, the Wine recognition, Ionosphere, ZOO and Pima data. Each dataset is permuted and randomly distributed in the sites. Different evaluation functions proposed by Ref. [19] are adapted for comparing the clustering results obtained from applying the two clustering algorithms on the test sets. The *F*-measure [31], Dunn Index [18] and Rand Index [31] are the three measures and their respective definitions also mentioned in Ref. [19] and each should be maximized. We have also analyzed the Inner Cluster variance—the sum of squared deviations between all data items of their associated cluster centre [19]. It should be minimized.

As mentioned in different publications about ant-based clustering methods, we must absolutely avoid complex parameter setting in order to simplify the use of this algorithm. The parameters that control the ants only are already numerous, without mentioning the coefficients dealing with the dissimilarity between different objects. Initially, the ants parameters were generated randomly within the bounds presented in the first applications (separately tested for tuning). These values will then be used in this paper for the tested datasets.

All runs have been performed for three different dissimilarity measures: Euclidean, Cosine, and Gower measures. All presented results have been averaged over 10 runs. Ants (10 agents) were simulated during 1,000,000 iterations when clustering objects. The number of agents should be kept small (for performance reasons), too many agents do not have any effect since agents walk in a randomly fashion, i.e., two agents coincide many times, over and over again, but they follow different walks.

The performance of a clustering algorithm can be judged with respect to its relative performance when compared to other algorithms. We therefore at the beginning choose the *k*-means algorithm. In our experiments, we run *k*-means algorithm using the correct cluster number *k*.

The following section presents in detail the conclusions draw from the experimental results with each of the tested datasets. In this paper we will only show some selected graphs to support our conclusions.

The results are mentioned in Tables 3–7. The tables show mean and standard deviations (in parentheses) for 1,000,000 runs, averaged over 10 runs. In experimental study we utilize the results reported in details in Ref. [32]. This big number of iterations is a common characteristic for different ant-based clustering algorithms.

The obtained partitions of ant clustering algorithms and statistics are very close to those of *k*-means on the analyzed datasets. The reader should keep in mind that, different from its competitor, ant-based clustering algorithms have not been provided with the correct number of clusters. We also observed the sensitivity to unequally sized clusters in analyzed datasets. We show the algorithms' performance on these datasets as reflected by *F*-measure.

Table 1
Results of evaluation functions on *k*-means, ACA and ACAM for square datasets

	ACA (Euc. m.)	ACA (cos. m.)	ACAM
square_1			
Clusters	4.720 (0.895)	4.560 (0.852)	4.000(0.200)
Rand Index	0.959 (0.020)	0.966 (0.187)	**0.985 (0.017)**
F-measure	0.944 (0.038)	0.951 (0.421)	**0.984 (0.023)**
Dunn Index	0.054 (0.023)	4.634 (2.772)	**0.9583 (1.997)**
Variance	5523.680 (375.048)	4.098 (1.034)	**1.290 (1.442)**
Class. err.	0.026 (0.005)	0.023 (0.036)	**0.018 (0.034)**
square_2			
Clusters	4.620 (1.112)	5.540 (0.921)	4.00 (0.283)
Rand Index	0.913 (0.061)	0.929 (0.197)	**0.969 (0.023)**
F-measure	0.886 (0.070)	0.885 (0.484)	**0.967 (0.031)**
Dunn Index	0.044 (0.015)	1.976 (1.707)	**6.901 (1.551)**
Variance	6580.113 (2920.295)	4.607 (1.408)	**1.853 (2.346)**
Class. err.	0.089 (0.097)	0.039 (0.1)	**0.036 (0.047)**
square_3			
Clusters	4.260 (0.795)	7.080 (1.181)	3.960 (0.280)
Rand Index	0.902 (0.039)	0.903 (0.197)	**0.948 (0.028)**
F-measure	0.878 (0.058)	0.846 (0.473)	**0.944 (0.038)**
Dunn Index	0.051 (0.017)	0.954 (0.469)	**6.314 (1.491)**
Variance	6446.134 (1686.293)	4.356 (0.948)	**2.232 (2.383)**
Class. err.	0.115 (0.081)	**0.056 (0.06)**	0.061 (0.055)
square_4			
Clusters	3.700 (0.700)	7.440 (1.169)	3.900 (0.361)
Rand Index	0.837 (0.081)	0.870 (0.174)	**0.912 (0.036)**
F-measure	0.814 (0.084)	0.791 (0.502)	**0.904 (0.046)**
Dunn Index	0.051 (0.015)	0.995 (0.334)	**5.581 (1.705)**
Variance	7091.038 (2546.104)	4.149 (1.261)	**3.394 (4.230)**
Class. err.	0.213 (0.122)	**0.094 (0.065)**	0.105 (0.071)

While the robust performance of the algorithms across a wide range of datasets has been demonstrated in these tables, our analysis in this report has focused in studying the scheme of adapting the α values that pose problems to ant clustering algorithms. Importantly, it must be noted that the cluster method is very sensitive to the choice of α and correlations over a specific thresholds are only achieve with the proper choice of α (see the performance of ACAM presented in Tables 1 and 2).

From some of results (see Table 1), the first ant-based algorithm ACA demonstrated to be incapable of correctly clustering the data in most simulations. The proposed algorithm, however, was capable of appropriately clustering the data in all runs (with strong correlations), but with varying numbers of clusters being found each time the algorithm was run. In almost all cases ACAM approach outperforms the results obtained by its competitor.

Despite the sufficient results presented here in first synthetic datasets, there are still several avenues for investigation that deserve to be pursued. For instance, because of too many clusters obtained by ACA, a hierarchical analysis of the datasets can be proposed by systematically varying some of the user-defined parameters: the use of set of objects (clusters) instead of a one object on a grid position scheme used here can be performed for an improvement.

Table 2
Results of evaluation functions on *k*-means, ACA and ACAM for square_5 and halfrings datasets

	ACA (Euc. m.)	ACA (cos. m.)	ACAM
square_5			
Clusters	4.060 (0.310)	4.720 (0.775)	4.140 (0.448)
Rand Index	0.962 (0.018)	0.929 (0.341)	**0.969 (0.012)**
F-measure	0.961 (0.026)	0.919 (0.477)	**0.970 (0.017)**
Dunn Index	0.065 (0.011)	2.328 (1.134)	**3.837 (0.657)**
Variance	5010.055 (603.425)	4.586 (1.158)	**1.301 (0.394)**
Class. err.	0.033 (0.013)	0.035 (0.043)	**0.028 (0.005)**
Halfrings			
Clusters	9.040 (1.509)	8.500 (0.900)	3.800 (0.980)
Rand Index	0.634 (0.043)	0.598 (0.176)	**0.701 (0.060)**
F-measure	0.522 (0.096)	0.469 (0.614)	**0.737 (0.082)**
Dunn Index	0.131 (0.033)	1.062 (0.454)	**1.858 (0.874)**
Variance	204.645 (81.438)	**3.951 (1.233)**	13.071 (29.328)
Class. err.	**0.010(0.003)**	0.087 (0.077)	0.119 (0.073)

Table 3
Results of evaluation functions on *k*-means, ACA and ACAM for Iris dataset

	k-means	ACA	ACAM
Iris 150			
Clusters	3.000	2.960	3.060 (0.420)
Rand Index	**0.824 (0.002)**	0.785 (0.022)	0.819 (0.015)
F-measure	**0.821 (0.003)**	0.773 (0.022)	0.810 (0.016)
Dunn Index	2.866 (0.188)	2.120 (0.628)	**2.959 (0.371)**
Variance	**0.861 (0.049)**	4.213 (1.609)	1.262 (0.961)
Class. err.	0.176 (0.004)	0.230 (0.053)	0.187 (0.040)
The best results (according to Rand Index)			
Clusters	3.000	3.000	3.000
Rand Index	0.829	0.814	**0.842**
F-measure	0.830	0.811	**0.842**
Dunn Index	2.939	2.306	**2.995**
Variance	**0.899**	1.486	0.914
Class. err.	0.167	0.187	**0.153**

The second type of analyzed data are as follows: Iris, Wine, Glass, ZOO and Pima datasets. The Iris datasets results are presented in Table 3. The ACAM approach outperforms the results obtained by ACA. Similarly to the results presented in the previous experiment, ant-based clustering algorithms consistently found almost always correct number of clusters with satisfying values of presented statistic measures.

Table 4 summarize the performance of ant-based clustering algorithms when applied to the Wine data. The best result presented in the context of Wine recognition belongs to the *k*-means algorithm. Dunn Index reached maximum value for the ACAM approach.

Table 5 shows the results for applying the ant-based algorithms in comparison to *k*-means to Ionosphere dataset as well as the best results according to Rand Index. It can be seen that these algorithms have very similar behaviors in most of the analyzed measures. Both ant-based clustering algorithms identify good number of clusters and ACAM obtain the smaller classification error than the *k*-means algorithm.

Table 6 depicts simulation results for ant clustering algorithm for ZOO dataset. It can be noted that it is difficult to choose appropriate similarity measures for all types of attributes. In this case ants found difficulties during Boolean-valued attributes comparison and the appropriate number of clusters is really difficult to obtain. For the ZOO dataset, the ant-based clustering

Table 4
Results of evaluation functions on *k*-means, ACA and ACAM for Wine dataset

	k-means	ACA	ACAM
Wine			
Clusters	3.000(0.000)	2.980 (1.140)	2.860 (0.347)
Rand Index	**0.909 (0.008)**	0.832 (0.021)	0.849 (0.051)
F-measure	**0.928 (0.007)**	0.855 (0.023)	0.868 (0.056)
Dunn Index	1.395 (0.022)	1.384 (0.101)	**1.407 (0.149)**
Variance	**6.290 (0.020)**	8.521 (0.991)	7.637 (2.859)
Class. err.	**0.071 (0.007)**	0.142 (0.030)	0.139 (0.082)
The best results (according to Rand Index)			
Clusters	3.000	3.000	3.000
Rand Index	**0.926**	0.872	0.914
F-measure	**0.943**	0.896	0.932
Dunn Index	1.327	**1.436**	1.399
Variance	**6.336**	8.157	6.435
Class. err.	**0.056**	0.101	0.067

Table 5
Results of evaluation functions on *k*-means, ACA and ACAM for Ionosphere dataset

	k-means	ACA	ACAM
Ionosphere			
Clusters	2.000 (0.000)	2.560 (0.535)	1.920 (0.271)
Rand Index	**0.578 (0.002)**	0.563 (0.017)	0.576 (0.012)
F-measure	0.705 (0.002)	0.676 (0.037)	**0.706 (0.007)**
Dunn Index	**1.211 (0.003)**	1.031 (0.198)	1.116 (0.329)
Variance	**23.167 (0.001)**	23.224 (2.224)	29.794 (21.627)
Class. err.	0.301 (0.002)	**0.300 (0.017)**	0.304 (0.017)
The best results (according to Rand Index)			
Clusters	2.000	2.000	2.000
Rand Index	0.582	0.586	**0.587**
F-measure	0.710	0.700	**0.715**
Dunn Index	1.212	0.841	**1.224**
Variance	**23.109**	23.743	23.221
Class. err.	0.296	0.291	**0.291**

Table 6
Results of evaluation functions on *k*-means, ACA and ACAM for ZOO dataset

	k-means	ACA	ACAM
ZOO			
Clusters	7.000 (0.000)	3.980 (0.616)	3.600 (0.632)
Rand Index	0.875 (0.036)	0.886 (0.036)	**0.889 (0.077)**
F-measure	0.747 (0.070)	0.764 (0.041)	**0.774 (0.072)**
Dunn Index	0.770 (0.222)	1.227 (0.273)	**1.391 (0.260)**
Variance	**1.645 (0.222)**	4.765 (1.089)	3.474 (1.897)
Class. err.	**0.160 (0.025)**	0.232 (0.040)	0.233 (0.068)
The best results (according to Rand Index)			
Clusters	7.000	4.000	4.000
Rand Index	**0.945**	0.930	0.943
F-measure	**0.893**	0.814	0.832
Dunn Index	0.810	1.491	**1.632**
Variance	**1.559**	3.792	2.415
Class. err.	**0.099**	0.178	0.178

algorithms demonstrated to be incapable of correctly grouping the data in most simulations. The results shown here depict that solutions obtained by ACA and ACAM have the same quality, maximizing the Rand Index, *F*-measure and Dunn Index in case of modified version ACAM.

The results presented in Table 7 suggests that these investigations are not very satisfying and the difficulties lies in the fact that the relationship between the attributes may not be directly detectable from their encoding, thus not presuming any metric relations even when the symbols represent similar items. Finally

Table 7
Results of evaluation functions on *k*-means, ACA and ACAM for Pima dataset

	k-means	ACA	ACAM
Pima			
Clusters	2.000 (0.000)	6.460 (1.590)	3.280 (1.510)
Rand Index	**0.560 (0.020)**	0.504 (0.013)	0.522 (0.022)
F-measure	**0.678 (0.029)**	0.473 (0.070)	0.574 (0.081)
Dunn Index	**0.983 (0.029)**	0.752 (0.140)	0.708 (0.290)
Variance	74.974 (1.835)	**45.226 (18.880)**	95.364 (60.665)
Class. err.	0.324 (0.023)	**0.321 (0.016)**	0.337 (0.013)
The best results (according to Rand Index)			
Clusters	2.000	5.000	2.000
Rand Index	**0.581**	0.536	**0.581**
F-measure	**0.709**	0.623	0.702
Dunn Index	**0.975**	0.776	0.842
Variance	73.808	**62.371**	97.946
Class. err.	**0.298**	0.331	**0.298**

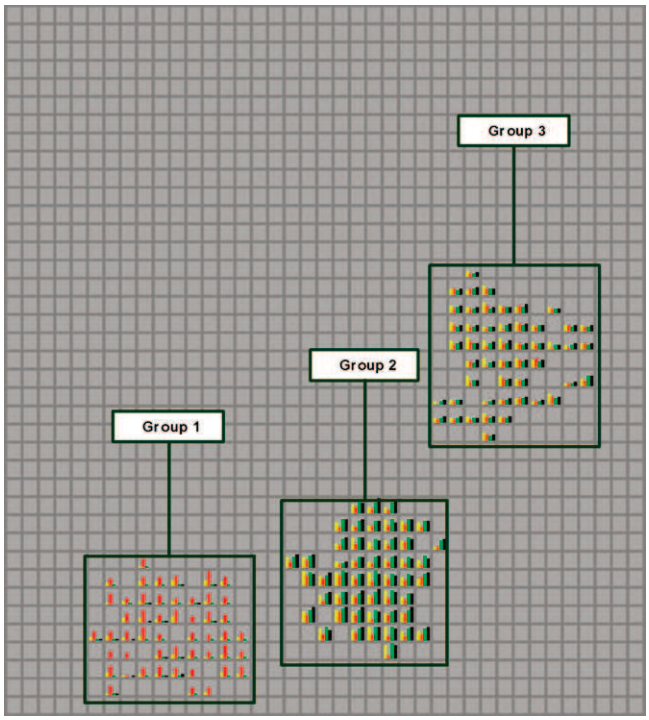

Fig. 2. Result of clustering (using Gower coefficient) for Iris 150 dataset.

the best performance of the ACAM presents the correct number of clusters obtained during this investigation.

The results obtained when different measures were used for decision making, show that the more suitable measure available to the agents, the better the performance. The results confirm the

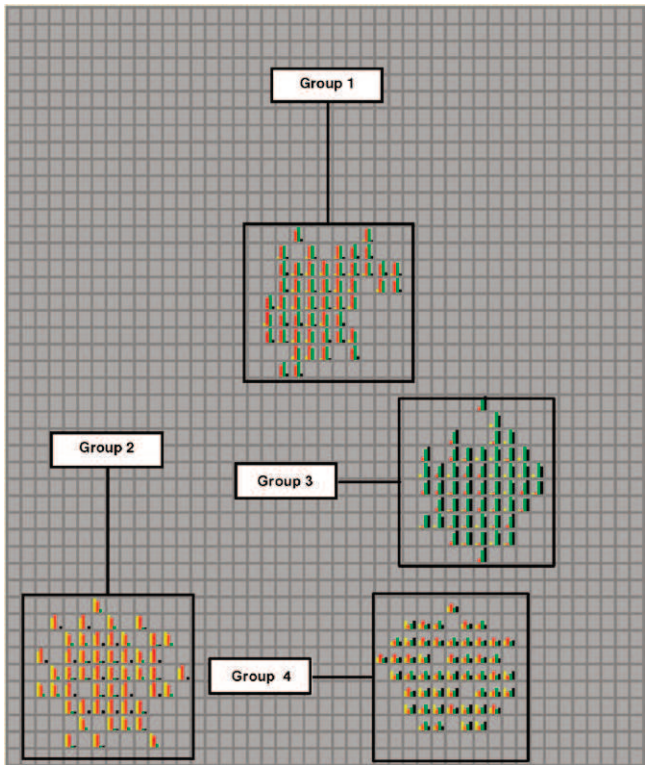

Fig. 3. Result of clustering for Iris 200 dataset (Euclidean measure).

intuition which says that binary representation of objects (in ZOO datasets) is really difficult for ant-based clustering algorithm, so it has to acquire more experiments with for example different methods of changing the parameter α.

The projection of data into a bi-dimensional output grid and position the items in neighbor regions is an advantage of these visual data exploration. The user can directly analyze the appropriate data clustering. Most importantly, ACA demonstrated a good robustness in terms of finding the correct number of clusters in some synthetic datasets, low variations of the results in terms of number of clusters found as well as number of objects within clusters (see also Iris datasets: Figs. 2 and 3).

To sum up, the proposed ant-based clustering algorithms have comparable accuracy in solutions almost for all cases and is significantly better in datasets with numerical attributes in solution accuracy than in datasets concerning ZOO, where the attributes are binary. It clearly shows that the objects in clusters are close to each other, but a small number of objects are grouped into a wrong cluster, suggesting the clustering results by ACA are less than satisfactory. To bring a matter to a satisfactory conclusion we must take into account different measures of dissimilarity or a standardization these values (especially for Cosine measure). There is however, an important drawback. The parameters of ants behavior needed to fine adapt during the performance of clustering. This is a consequence of the lack of understanding of the impact in the global behavior of a colony of simulated insect-like agents.

7. Conclusions

We have presented in this paper a new ant-based clustering algorithm called ACA and its modification for data clustering in a knowledge discovery context. ACA introduces new ideas and modifications in Lumer and Faieta's algorithm in order to improve the convergence. The main features of this algorithm are the following ones. ACA deals with numerical databases. It does not require to establish the number of clusters or any information about the feature of the clusters.

It can be noted that the appropriate chosen dissimilarity measure generates a much correct number of clusters, in most synthetic datasets, the correct number of groups of data are generated. It can also be spotted from presented results that the adaptation scheme of parameter α tend to be better adapted in modified version of ACAM. This is a first step towards that goal. With this knowledge and considering the characteristics of a particular problem, we could obtain good results of clustering but in the future we try to improve the performance of the algorithm. Two future research directions can be identity. Firstly, the proposed ant-based clustering algorithm can only be applied to the clustering problems with numerical (or binary) attributes so the next step concerns the nominal attributes mixed with numerical attributes. Furthermore, a hybrid method by combining the ant-based clustering algorithm ACA with other metaheuristics algorithm is also deserved to develop. We have employed simulated annealing to determine the scheme of α-adaptation and we investigated the better robustness of this approach. Subsequently we can use this values to conduct the adaptive clustering more precisely. Future work should be focused on studying the effects of using different communication strategies via pheromone in these approaches. We also need to eliminate the bias on dissimilarity measures provoked by different scales within data attributes, we should standardize the database and try to find the best metric. Future work consists also in testing how this model with new ideas of learning process via pheromone updating rules scales with large databases. We are also considering other biological inspirations from real ants for analysis a clustering problem, for example learning the template and other principles of recognition system.

References

[1] M. Berger, I. Rigoutsos, An algorithm for point clustering and grid generation, IEEE Trans. Syst. Man Cybern. 21 (5) (1991) 1278–1286.
[2] E. Bonabeau, From classical models of morphogenesis to agent-based models of patern formation, Artif. Life 3 (1997) 191–209.
[3] E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm Intelligence. From Natural to Artificial Systems, Oxford University Press, New York, 1999.
[4] M.R. Brito, E. Chavez, A. Quiroz, J. Yukich, Connectivity of the mutual *k*-nearest-neighbor graph for clustering and outlier detection, Stat. Prob. Lett. 35 (1997) 33–42.
[5] P. Brucker, On the complexity of clustering problems, in: R. Henu, B. Korte, W. Oetti (Eds.), Optimization and Operations Research, Springer–Verlag, 1978, pp. 45–54.
[6] L. Chretien, Organisation Spatiale du Materiel Provenant de L'excavation du nid chez Messor Barbarus et des Cadavres d'ouvrieres chez *Lasius niger* (Hymenopterae: Formicidae), PhD thesis, Universite Libre dr Bruxelles, 1996.
[7] S.C. Chu, J.F. Roddick, J.S. Pan, A comparative study and extensions to *k*-medoids algorithms, in: Proceedings of the Fifth International Conference on Optimization: Techniques and Applications, Hong Kong, China, 2001.
[8] J.-L. Deneubourg, S. Goss, N. Franks, A. Sendova-Franks, C. Detrain, The dynamics of collective sorting: robot-like ant and ant-like robot, in: Proceedings of the First Conference on Simulation of Adaptive Behavior: From Animals to Animats, MIT Press, 1991, pp. 356–365.
[9] J.-L. Deneubourg, S. Goss, N. Franks, A. Sendova-Franks, C. Detrain, L. Chretien, The dynamics of collective sorting: robot-like ant and ant-like robot, in: J.A. Meyer, S.W. Wilson (Eds.), Proceedings of the First Conference on Simulation of Adaptive Behavior. From Animals to Animats, 1991, pp. 356–365.
[10] M. Dorigo, T. Stützle, Ant Colony Optimization, MIT Press, Cambridge, 2004.
[11] R. Dubes, A. Jain, Clustering methodologies in exploratory data analysis, in: M. Yovits (Ed.), Advances in Computers, vol. 19 of Advances in Computers, Academic Press, New York, 1980.
[12] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: E. Simuoudis, J. Han, U. Fayyard (Eds.), Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, AAAI Press, Portland, USA, 1996 , pp. 226–231.
[13] N.R. Franks, A.B. Sendova-Franks, Brood sorting by ants: distributing the workload over the work surface, Behav. Ecol. Sociobiol. 30 (1992) 109–123.
[14] V. Ganti, J. Gehrke, R. Ramakrishnan, Cactus-clustering categorical data using summaries, in: Proceedings of the International Conference on Knowledge Discovery and Data Mining, San Diego, USA, (1999), pp. 73–83.
[15] D. Gibson, J. Kleinberg, P. Raghavan, Clustering categorical data: an approach based on dynamical systems, in: Proceedings of the 24th VLDB Conference, 1998, pp. 311–322.
[16] S. Guha, R. Rastogi, K. Shim, Cure: an efficient clustering algorithm algorithm for large databases, in: Proceedings of the ACM SIGMOD International Conference on the Management of Data, Seatle, USA, (1998), pp. 73–84.
[17] H. Gutowitz, Complexity-seeking ants, Unpublished report, 1993.
[18] M. Halkidi, M. Vazirgiannis, I. Batistakis, Quality scheme assesment in the clustering process, in: Proceedings of the Fourth European Conference on Principles of Data Mining and Knowledge Discovery, vol. 1910, Springer–Verlag of LNCS, 2000, pp. 265–267.
[19] J. Handl, J. Knowledge, M. Dorigo, Ant-based clustering: a comparative study of its relative performance with respect to *k*-means, average link and id-som, Technical Report 24, IRIDIA, Universite Libre de Bruxelles, Belgium, 2003.
[20] J. Handl, B. Meyer, Improved ant-based clustering and sorting in a document retrieval interface, in: S. Verlag (Ed.), Proceedings of the Seventh international Conference on Parallel Problem Solving from Nature, PPSN-VII, LNCS, Berlin, 2002, pp. 913–923.
[21] P. Hansen, B. Jaumard, Minimum sum of diameter clustering, J. Classif. 4 (2) (1987).
[22] A. Jain, R. Dubes, Algorithms for Clustering Data, Prentice Hall, New Jersey, 1998.
[23] G. Karypis, E.-H. Han, V. Kumar, Chameleon: a hierarchical clustering algorithm using dynamic modeling, Computer 32 (1999) 32–68.
[24] L. Kaufman, P. Russeeuw, Finding groups in data: an introduction to cluster analysis, John Wiley and Sons, 1990.
[25] J. Kennedy, R.C. Eberhart, Swarm Intelligence, Morgan Kaufmann, San Francisco, 2001.
[26] R. Lee, Clustering analysis and its applications, Adv. Inform. Syst. 8 (1981) 169–292.
[27] E. Lumer, B. Faieta, Diversity and adaptation in populations of clustering ants, in: Proceedings of the Third international Conference on Simulation of Adaptive Behavior: From Animals to Animats, vol. 3, MIT Press, Cambridge, 1994, pp. 489–508.
[28] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of the Fifth Berkeley Symposium on Mathematics, Statistics and Probability, 1967, pp. 281–296.

[29] R. Ng, J. Han, Efficient and effective clustering methods for spatial data mining, in: J.B. Bocca, M. Jarke, C. Zaniolo (Eds.), Proceedings of the Twentieth International Conference on Very Large Data Bases, Morgan Kaufmann, Santiago, Chile, 1994, pp. 144–155.
[30] S.A. Oprisan, V. Holban, B. Moldoveanu, Functional self-organisation performing wide-sense stochastic processes, Phys. Lett. A 216 (1996) 303–306.
[31] C.V. Rijsbergen, Information Retrieval, 2nd edition, Butterworth, London, 1979.
[32] R. Skinderowicz, Zastosowanie algorytmow mrowkowych do grupowania danych, Master's thesis, Institute of Computer Science, University of Silesia, 2007.
[33] W.J. Welch, Algorithmic complexity; three np-hard problems in computational statistics, J. Stat. Comput. 15 (1) (1997).
[34] M. Zait, H. Messatfa, A comparative study of clustering methods, FGCS J., Special Issue Data Min. (1997).