Control and Cybernetics

vol. **39** (2010) No. 2

Ant colony metaphor in a new clustering algorithm[*]

by

Urszula Boryczka

Institute of Computer Science, University of Silesia, Sosnowiec, Poland

Abstract: 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. Clustering with swarm–based algorithms is emerging as an alternative to more conventional clustering methods, such as k–means algorithm. This proposed approach mimics the clustering behavior observed in real ant colonies.

As a case study, this paper focuses on the behavior of clustering procedures in this new approach. The proposed algorithm is evaluated on a number of well–known benchmark data sets. Empirical results clearly show that the ant clustering algorithm (*ACA*) performs well when compared to other techniques.

Keywords: data mining, cluster analysis, ant clustering algorithm.

1. Introduction

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. Clustering algorithms can be classified into partitioning and hierarchical algorithms. Partitioning algorithms create a partitioning of objects into a set of clusters. Hierarchical algorithms construct a hierarchical decomposition of the set of objects. The hierarchical decomposition is represented by a tree strategy that separates the objects into small subsets until each consists only of sufficiently similar objects. There exists a large number of clustering algorithms in the literature including k–means (MacQueen, 1967), k-medoids (Kaufman and Russeeuw, 1990), *CACTUS* (Ganti, Gehrke and Ramakrishna, 1999), *CURE* (Guha, Rastogi and

[*]Submitted: January 2009; Accepted: October 2009.

Shim, 1998), *CHAMELEON* (Karypis, Han and Kumar, 1999) and *DBSCAN* (Ester et al., 1996). No single algorithm is suitable for all types of problems, however, the k–medoids algorithms have been shown (Kaufman and Russeeuw, 1990) to be robust to outliers, compared with centroid–based clustering. The drawback of the k–medoids algorithm is the time complexity of determining the medoids. In this paper, a novel ant–based clustering algorithm (ACA) is proposed to improve the performance of many k–medoids–based algorithms. A new version of ACA algorithm is inspired by the behavior of real ants. The paper is organized as follows: Section 2 gives a detailed description of the biological inspirations and first experiments. Section 3 presents the algorithm. Section 4 presents the experiments that have been conducted to set the parameters of ACA regardless of the data sets. The last section concludes and discusses future evolution of ACA.

2. Biological inspirations and algorithms

Clustering and sorting behavior of ants has stimulated research in design of new algorithms for data analysis and partitioning. Several species of ants cluster corpses 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 (Bonabeau, Dorigo and Theraulaz, 1999).

In several species of ants, workers have been reported to form piles of corpses — cemeteries — to clean the nests. Chretien (1996) 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 Deneubourg et al. (1991). Brood sorting was observed by Franks and Sendova-Franks (1992) in the ant *Leptothorax unifasciatus*. Workers of this species gather the larvae according to their size. Franks and Sendova-Franks (1992) have intensively analyzed the distribution of brood within the brood cluster.

Deneubourg et al. (1991) have proposed two closely related models to account for the two above–mentioned phenomena of corpse clustering and larval sorting in ants. General idea is that isolated items should be picked up and dropped at some other location where more items of that type are present. 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

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

where:

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

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.

Franks and Sendova-Franks (1992) 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 the last T time units, divided by the largest possible number of items that can be encountered during this time.

Gutowitz (1993) suggested the use of spatial entropy to track the dynamics of clustering. 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, Holban and Moldoveanu (1996) proposed a variant of the Deneubourg basic model (hereafter called BM), in which the influence of previously encountered objects is discounted by a time factor.

Bonabeau (1997) also explored the influence of various weighting functions, especially those with short–term activation and long–term inhibition.

Lumer and Faieta (1994) 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 (herafter 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 site 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

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

where:

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

Lumer and Faieta (1994) define picking up and dropping probabilities as follows:

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

$$p_d(o_i) = \begin{cases} 2f(o_i) & \text{when } f(o_i) < k_2 \\ 1, & \text{when } f(o_i) \geq k_2 \end{cases} \tag{1}$$

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:

Algorithm 1: The Lumer–Faieta algorithm

```
/***Initialization***/
**for** every object $o_i$ **do**
  Place $o_i$ randomly on grid
**end for**
**for** all ants **do**
  place ant at randomly selected site
**end for**
{***main loop***}
**for** all ants **do**
  **for** $t = 1$ **to** $t_{max}$ **do**
    **if** ((agent unladen) and (site occupied by item $o_i$)) **then**
      Compute $F(o_i)$ and $p_p(o_i)$
      Draw random real number $R \in (0, 1)$
      **if** ($R \leq p_p(o_i)$) **then**
        Pick up item $o_i$
      **end if**
    **else**
      **if** (agent carrying item $o_i$) and (site empty)) **then**
        Compute $f(o_i)$ and $p_d(o_i)$
        Draw random real number $R \in (0, 1)$
        **if** ($R \leq p_d(o_i)$) **then**
          Drop item
        **end if**
      **end if**
    **end if**
    Move to randomly selected neighboring site not occupied by other agent
  **end for**
**end for**
Print location of items.
```

3. Ant Clustering Algorithm — *ACA*

The ant 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 (2002) extended Lumer and Faieta's algorithm and proposed an application to 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.

We have employed a modified version of the „short–term memory" introduced by Lumer and Faieta (1994). Each ant has a permission to exploit its memory according to the following rules: if an ant is situated at grid cell p and carries 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 neighbourhood 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:

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

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

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

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

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

1. Independent of the data.
2. Being a function of the size of the data set.

The first group includes:

- the number of agents, which is set to be 10,
- the size of the agents' short–term memory, which we also set at 10,

- the initial clustering phase (from t_{start} to t_{end}: $t_{start} = 0.45 \cdot N$, $t_{end} = 0.55 \cdot N$, where N denotes the number of iterations),
- we replace the scaling parameter $\frac{1}{\sigma^2}$ by $\frac{1}{N_{occ}}$ after the initial clustering phase, where N_{occ} is the actual observed number of occupied grid cells within the local neighbourhood.

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

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

- using a square grid with resolution of $\sqrt{10 \cdot N_{max}} \times \sqrt{10 \cdot N_{max}}$,
- the step permitting sampling of each possible grid position within one move, which is obtained by setting it to step size: $\sqrt{20 \cdot N_{max}}$,
- the number of iterations: $\sqrt{2000 \cdot N_{max}}$, 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 α prevents the formation of clusters on the grid; on the other hand, a too large 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 heterogeneous population of ants is used — 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 $r_{fail} = \frac{N_{fail}}{N_{active}}$ where N_{active} is fixed to 100. The agent's parameter α is then updated using the rule:

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

High–level description of the ant clustering algorithm is presented below:

Algorithm 2: ACA algorithm

```
0 /*Initialization Phase*/
1 Randomly scatter $o_i$ object on the grid file
2 for each agent $a_j$ do
3     random_select_object ($o_i$)
4     pick_up_object $o_i$
5     place_agent $a_j$ at randomly selected empty grid location
6 end for
7 {*Main loop*}
8   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 $o$
            Compute $f^*(o_i)$ and $p^*_{pick}(o_i)$
            Pick_up_object $o_i$
         **end while**
      **end if**
   **end for**
**end**
```

4. Experimental results

The performance of the clustering algorithm may be judged with respect to its relative performance when compared to other algorithms. For this purpose, at the beginning we chose the k–means algorithm. In our experiments, we ran k–means algorithm using the correct cluster number k.

In order to evaluate the resulting partitions obtained by ACA we have set up the following method. The first data sets used to illustrate the performance of the algorithms were a modified version of the well–known data sets proposed to study the standard ant–based clustering algorithm (Handl, Knowles and Dorigo, 2003). The Square data sets are the most popularly used type of data sets. They are two–dimensional and consist of four clusters arranged as a square. To conform to distributed data sets the data are spread uniformly among the various sites.

Our analysis in this report has focused on studying the scheme of adapting the α values that pose problems to ant clustering algorithms. Importantly, it must be noted that the clustering method is very sensitive to the choice of α and correlations over a specific thresholds are only achieved with the proper choice of α (see the performance of ACA presented in Tables 1 and 2). The parameter α weights the influence of the distance measure in determining the clusters. ACA performs satisfactorily on all six data sets, in fact it is hardly affected at all by the increasing deviations between cluster sizes (especially for the Cosine measure). The results demonstrate that, if clear cluster structures exist within the data, the ant clustering algorithm is quite reliable at identifying the correct number of clusters. This is an indication that the structure within the data is not easily pronounced.

Table 1. Evaluation of results of the *ACA* (with different dissimilarity measures) for Square datasets.

square_1	*ACA (Euc. m.)*	*ACA (cos. m.)*
Clusters	4.720 (0.895)	4.560 (0.852)
Rand Index	0.959 (0.020)	0.966 (0.187)
F–measure	0.944 (0.038)	0.951 (0.421)
Dunn Index	0.054 (0.023)	4.634 (2.772)
Variance	5523.680 (375.048)	4.098 (1.034)
Class. err.	0.026 (0.005)	0.023 (0.036)
square_2	*ACA (Euc. m.)*	*ACA (cos. m.)*
Clusters	4.620 (1.112)	5.540 (0.921)
Rand Index	0.913 (0.061)	0.929 (0.197)
F–measure	0.886 (0.070)	0.885 (0.484)
Dunn Index	0.044 (0.015)	1.976 (1.707)
Variance	6580.113 (2920.295)	4.607 (1.408)
Class. err.	0.089 (0.097)	0.039 (0.1)
square_3	*ACA (Euc. m.)*	*ACA (cos. m.)*
Clusters	4.260 (0.795)	7.080 (1.181)
Rand Index	0.902 (0.039)	0.903 (0.197)
F–measure	0.878 (0.058)	0.846 (0.473)
Dunn Index	0.051 (0.017)	0.954 (0.469)
Variance	6446.134 (1686.293)	4.356 (0.948)
Class. err.	0.115 (0.081)	0.056 (0.060)
square_4	*ACA (Euc. m.)*	*ACA (cos. m.)*
Clusters	3.700 (0.700)	7.440 (1.169)
Rand Index	0.837 (0.081)	0.870 (0.174)
F–measure	0.814 (0.084)	0.791 (0.502)
Dunn Index	0.051 (0.015)	0.995 (0.334)
Variance	7091.038 (2546.104)	4.149 (1.261)
Class. err.	0.213 (0.122)	0.094 (0.065)

Table 2. Evaluation of results of the *ACA* (with different dissimilarity measures) for Square datasets.

square_5	*ACA (Euc. m.)*	*ACA (cos. m.)*
Clusters	4.060 (0.310)	4.720 (0.775)
Rand Index	0.962 (0.018)	0.929 (0.341)
F–measure	0.961 (0.026)	0.919 (0.477)
Dunn Index	0.065 (0.011)	2.328 (1.134)
Variance	5010.055 (603.425)	4.586 (1.158)
Class. err.	0.033 (0.013)	0.035 (0.043)
halfrings	*ACA (Euc. m.)*	*ACA (cos. m.)*
Clusters	9.040 (1.509)	8.500 (0.900)
Rand Index	0.634 (0.043)	0.598 (0.176)
F–measure	0.522 (0.096)	0.469 (0.614)
Dunn Index	0.131 (0.033)	1.062 (0.454)
Variance	204.645 (81.438)	**3.951 (1.233)**
Class. err.	**0.010 (0.003)**	0.087 (0.077)

We have also applied *ACA* to the real world databases from the Machine Learning repository, which are often used as benchmarks. It is useful to show experimentally the efficiency of *ACA* on data with known properties and difficulty. The real data collections used were the Iris data, the Wine Recognition, Ionosphere and Pima data. Each dataset was permuted and randomly distributed in the sites. Different evaluation functions, proposed by Handl, Knowles and Dorigo (2003) are adapted for comparing the clustering results obtained from applying the two clustering algorithms on the test sets. The F–measure (Rijsbergen, 1979), Dunn Index (Halkidi, Vazirgiannis and Batistakis, 2000) and Rand Index (Rijsbergen, 1979) are the three measures and their respective definitions also given in Handl, Knowles and Dorigo (2003), 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 (Handl, Knowles and Dorigo, 2003). It is to be minimized.

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 results are provided in Tables 3 through 6. The tables show mean and standard deviations (in brackets) for 1,000,000 runs, averaged over 10 runs. The results of the experimental study are reported in details in Skinderowicz

(2007). The so high 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 approach on the analyzed data sets. 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 data sets. We show the algorithms' performance on these data sets as reflected by F–measure.

The Iris data sets results are presented in Table 3. The k–means approach outperforms the results obtained by ACA. Similarly to the results presented in the previous experiment, the ant–based clustering algorithm consistently found almost always the correct number of clusters with satisfying values of statistical measures.

Table 4 summarizes the performance of the ant–based clustering algorithm when applied to the Wine data. The best result presented in the context of Wine recognition belongs to the k–means algorithm. Classification error reached maximum value for the ACA approach, equal to 0.142.

Table 5 shows the results for applying the ant–based algorithms in comparison to k–means for the Ionosphere data set as well as the best results according to the Rand Index. It can be seen that these algorithms have very similar behavior in most of the analysed measures. Both algorithms identify good number of clusters and ACA yields a smaller classification error than the k–means algorithm.

The results presented in Table 6 suggest that these investigations are not very satisfying and the difficulties lie 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 (Variance). Finally, the good performance of the ACA presents the correct number of clusters obtained during this investigation (Classification error).

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 is. The results confirm the intuition which says that binary representation of objects (in some data sets) is really difficult for ant–based clustering algorithm. In this case the algorithm needs more experiments with different methods of changing the parameter α.

The projection of data into a bi–dimensional output grid and position the items in neighboor regions gives an advantage of the visual data exploration (see Fig. 1). By doing this, the algorithm is capable of clustering together objects that are similar to each other and presenting the result of this process on a bi–dimensional display that can be easily inspected visually helping the user to deal with the overload of information. The advantage of the visual data exploration is that the user is directly involved in the data mining process.

Most importantly, ACA demonstrated good robustness in terms of finding the correct number of clusters in some synthetic data sets, low variations of

Table 3. Evaluation of results of the k–means and *ACA* algorithms for the Iris dataset.

Iris 150	*k–means*	*ACA*
Clusters	3.000	2.960
Rand Index	0.824 (0.002)	0.785 (0.022)
F–measure	0.821 (0.003)	0.773 (0.022)
Dunn Index	2.866 (0.188)	2.120 (0.628)
Variance	0.861 (0.049)	4.213 (1.609)
Class. err.	0.176 (0.004)	0.230 (0.053)
The best results (according to Rand Index)		
Clusters	3.000	3.000
Rand Index	0.829	0.814
F–measure	0.830	0.811
Dunn Index	2.939	2.306
Variance	0.899	1.486
Class. err.	0.167	0.187

Table 4. Evaluation of results of the k–means and *ACA* algorithms for the Wine dataset.

Wine	*k–means*	*ACA*
Clusters	3.000 (0.000)	2.980 (1.140)
Rand Index	0.903 (0.008)	0.832 (0.021)
F–measure	0.928 (0.007)	0.855 (0.023)
Dunn Index	1.395 (0.022)	1.384 (0.101)
Variance	6.290 (0.020)	8.521 (0.991)
Class. err.	0.071(0.007)	0.142 (0.030)
The best results (according to Rand Index)		
Clusters	3.000	3.000
Rand Index	0.926	0.872
F–measure	0.943	0.896
Dunn Index	1.327	1.436
Variance	6.336	8.157
Class. err.	0.056	0.101

Table 5. Evaluation of results of the k–means and *ACA* algorithms for the Ionosphere dataset.

Ionosphere	*k–means*	*ACA*
Clusters	2.000 (0.000)	2.560 (0.535)
Rand Index	0.578 (0.002)	0.563 (0.017)
F–measure	0.705 (0.002)	0.676 (0.037)
Dunn Index	1.211 (0.003)	1.031 (0.198)
Variance	23.167 (0.001)	23.224 (2.224)
Class. err.	0.301(0.002)	0.300 (0.017)
The best results (according to Rand Index)		
Clusters	2.000	2.000
Rand Index	0.582	0.586
F–measure	0.710	0.700
Dunn Index	1.212	0.841
Variance	23.109	23.743
Class. err.	0.296	0.291

Table 6. Evaluation of results of the k–means and *ACA* algorithms for the Pima dataset.

Pima	*k–means*	*ACA*
Clusters	2.000 (0.000)	6.400 (1.590)
Rand Index	0.960 (0.020)	0.504 (0.013)
F–measure	0.678 (0.029)	0.473 (0.070)
Dunn Index	0.983 (0.029)	0.752 (0.140)
Variance	74.974 (1.835)	45.226 (18.880)
Class. err.	0.324 (0.023)	0.321 (0.016)
The best results (according to Rand Index)		
Clusters	2.000	5.000
Rand Index	0.581	0.536
F–measure	0.709	0.623
Dunn Index	0.975	0.776
Variance	73.808	62.971
Class. err.	0.278	0.331

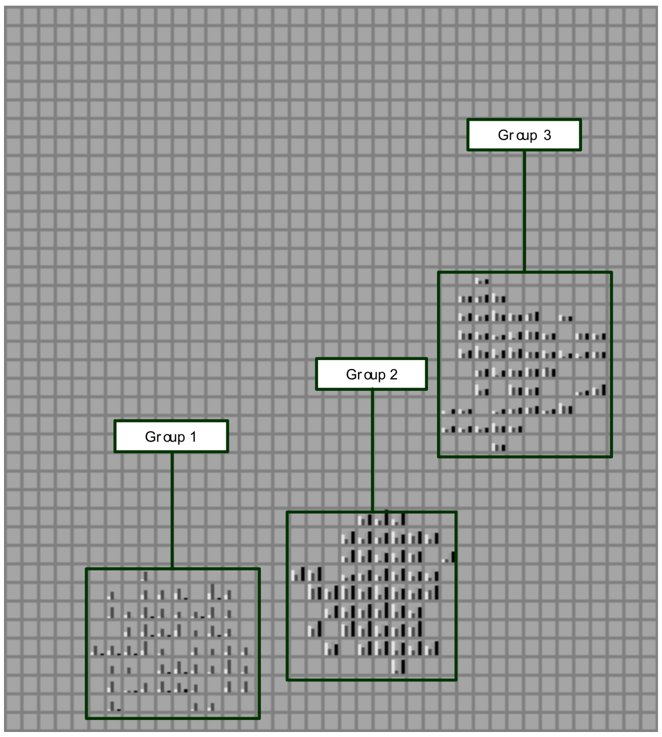

Figure 1. Visualization of clustering for the Iris data set (150 objects)

the results in terms of the number of clusters found as well as the number of objects within clusters (see also: Iris data set). *ACA* does not need the number of clusters to proceed with the clustering task and the results obtained by the algorithm are similar or even better than those by k–means approach for some of the metrics considered in this work.

To sum up, the proposed ant–based clustering algorithm has comparable accuracy in solutions for almost all cases and is significantly better in data sets with numerical attributes in solution accuracy than in data sets concerning binary attributes. 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 that 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 standarization of these values (especially for the Cosine measure). There is, however, an important drawback. The parameters of ant behavior needed to be fine-tuned during the performance of clustering. This is a consequence of the lack of understanding of the global behavior of a colony of simulated insect–like agents.

Following the conclusions from the results presented here, 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 data sets can be proposed by systematically varying some of the user–defined parameters: the use of set of objects (clusters) instead of one object on a grid position scheme used here can be performed for an improvement.

5. Conclusions

In this paper, we have presented a new ant clustering algorithm called ACA, 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 establishing the number of clusters nor any information about the feature of the clusters.

The ant clustering algorithm has a number of properties that make out of it an interesting candidate for improvement in the context of applications. Firstly, because of its linear scaling behavior it is attractive for use in large data sets, e.g. in information retrieval systems. Secondly — this algorithm deals with the outliers within data sets. In addition, the ant clustering algorithm is capable to analyse different kinds of data, which can be divided into clusters of the hardly anticipated shapes on the grid files.

The scheme of α–adaptation, proposed originally by J. Handl, is not as good as we assumed in our approach. This scaling parameter plays an important role in the clustering process, so the changing scheme of its values should be strongly connected to the effectiveness of the algorithm. This parameter is responsible for the cluster number. If the clusters on a few hierarchical levels exists, this version of the ant clustering algorithm will identify the high level connections, so the generated clusters could be recursively processed.

Future work consists 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 analysing the clustering problem, for example learning the template and other principles of recognition systems.

References

Bonabeau, E. (1997) From classical models of morphogenesis to agent–based models of pattern formation. *Artificial Life*, **3**, 191–209.

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

Chretien, L. (1996) 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, Université Libre de Bruxelles.

Deneubourg, J.-L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C. and Chretien, L.(1991) The dynamics of collective sorting: Robot–like ant and ant–like robot. In: J.A. Meyer and S.W. Wilson, eds., *First Conference on Simulation of Adaptive Behavior. From Animals to Animats*, 356–365.

Ester, M., Kriegel, H.-P., Sander, J. and Xu, X. (1996) A density–based algorithm for discovering clusters in large spatial databases with noise. In: E. Simuoudis, J. Han and U. Fayyard, eds., *Second International Conference on Knowledge Discovery and Data Mining*, AAAI Press, Portland, USA, 226–231.

Franks, N.R. and Sendova-Franks, A.B. (1992) Brood sorting by ants: Distributing the workload over the work surface. *Behav. Ecol. Sociobiol.*, **30**, 109–123.

Ganti, V., Gehrke, J. and Ramakrishna, R. (1999) Cactus–clustering categorical data using summaries. In: *International Conference on Knowledge Discovery and Data Mining*, San Diego, USA, 73–83.

Guha, S., Rastogi, R. and Shim, K. (1998) Cure: an efficient clustering algorithm algorithm for large databases. In: *ACM SIGMOD International Conference on the Management of Data*, Seatle, USA, 73–84.

Gutowitz, H. (1993) Complexity – Seeking Ants. Unpublished report.

Halkidi, M., Vazirgiannis, M. and Batistakis, I. (2000) Quality scheme assesment in the clustering process. In: *Proceedings of the Fourth European Conference on Principles of Data Mining and Knowledge Discovery.* **LNCS 1910**, Springer Verlag, 265–267.

Handl, J. and Meyer, B. (2002) Improved ant–based clustering and sorting in a document retrieval interface. In: *PPSN — VII. Seventh international Conference on Parallel Problem Solving from Nature*, **LNCS 2439**, Berlin, 913–923.

Handl, J., Knowles, J. and Dorigo, M. (2003) Ant–based clustering: a comparative study of its relative performance with respect to k–means, average link and id–som. *Technical Report* **24**, IRIDIA, Université Libre de Bruxelles, Belgium.

Karypis, G., Han, E.-H. and Kumar, V. (1999) Chameleon: a hierarchical clustering algorithm using dynamic modeling. *Computer* **32**, 32–68.

Kaufman, L. and Russeeuw, P. (1990) *Finding Groups in Data: An Introduction to Cluster Analysis.* John Wiley and Sons.

Lumer, E. and Faieta, B. (1994) Diversity and adaptation in populations of clustering ants. In: *Third Intern. Conference on Simulation of Adaptive Behavior: From animals to Animats 3.* MIT Press, Cambridge, 489–508.

MacQueen, J. (1967) Some methods for classification and analysis of multivariate observations. In: *5th Berkeley Symposium on Mathematics, Statistics and Probability*, 281–296.

OPRISAN, S.A., HOLBAN, V. and MOLDOVEANU, B. (1996) Functional self–organisation performing wide–sense stochastic processes. *Phys. Lett.* **A 216**, 303–306.

RIJSBERGEN, C.V. (1979) *Information Retrieval*, 2nd edition. Butterworth, London.

SKINDEROWICZ, R. (2007) *Zastosowanie algorytmow mrowkowych do grupowania danych.* (Application of ant algorithms to data grouping; in Polish). Master's thesis, Institute of Computer Science, University of Silesia.