Chapter 16
A Multiagent, Multiobjective Clustering Algorithm

Daniela S. Santos, Denise de Oliveira, and Ana L. C. Bazzan

Abstract This chapter presents MACC, a multi ant colony and multiobjective clustering algorithm that can handle distributed data, a typical necessity in scenarios involving many agents. This approach is based on independent ant colonies, each one trying to optimize one particular feature objective. The multiobjective clustering process is performed by combining the results of all colonies. Experimental evaluation shows that MACC is able to find better results than the case where colonies optimize a single objective separately.

16.1 Introduction

Clustering is widely used in data mining to separate a data set into groups of similar objects. The importance of clustering is clear in applications related to biology, social sciences, computer science, medicine, and so on. Consequently, many clustering methods have already been developed. One issue with most of these methods is that they rely on central data structures. However, the current use of Internet resources (distribution of data, privacy, etc.) requires new ways of dealing with data clustering. This meets the recent trend around the integration of agent technologies and data mining (see [1] and references therein). In this publication, the authors identify and discuss two main challenges concerning the integration of agents and data mining, namely *data mining driven agent learning* and *agent driven data mining*. In the present chapter we address the latter by proposing the Multi Ant Colony Clustering algorithm (MACC). This is also the line followed in [2] where agent-based data mining was used to integrate knowledge and facilitate the annotation of proteins. Our work thus has addressed some topics listed in [1] such as agent-based distributed data mining, multi-data source mining, and distributed agent-based data gathering and processing. While in [2] we have addressed supervised learning, here we turn to clustering.

Daniela S. Santos, Denise de Oliveira, and Ana L. C. Bazzan
Instituto de Informática, UFRGS
C.P. 15064, 91501-970, P.Alegre, RS, Brazil
e-mail: \{daniela.scherer, denise.oliveira, bazzan\}@inf.ufrgs.br

L. Cao (ed.), Data Mining and Multi-agent Integration, DOI: 10.1007/978-1-4419-0522-2_16,

Clustering methods differ not only in many of their most basic proprieties, such as the data type handled, but also in the form of the final partitioning, in the assumptions about the shape of the clusters, and in the parameters that have to be provided. Each clustering algorithm looks for clusters according to a different criterion. This can be a problem because many data sets present different shapes and size of clusters that a single objective clustering is not able to reveal. Moreover, the same data can have more than one relevant structure, each one related to a different cluster definition or to a different refinement level [3].

Recently, several biologically inspired algorithms have been introduced to solve the clustering problem [4, 5, 6, 7, 8]. These algorithms are characterized by the interaction of a large number of simple agents that interact in a multiagent system. These agents can perceive and change their environment locally and they are inspired by ant colonies, flocks of birds, swarms of bees, etc. However, these algorithms have been focussing on single objective clustering.

As an effort to overcome some limitations of single objective clustering algorithms, multiobjective clustering algorithms such as [3, 9] have been proposed. The basic idea of the multiobjective approach is to optimize more than one objective in the same clustering. Using this approach we can find different shapes and sizes of clusters and different types of structures in a data set.

MACC, the algorithm we propose, is inspired by ant colony optimization (ACO) and multiobjective clustering. The central idea of MACC is to simultaneously use several ant colonies, each colony aiming to optimize one objective. In this particular work we focus on two objectives in order to compare our results with previous works on multiobjective clustering. This way, one colony minimizes the compactness, while the other maximizes the connectivity of clusters. The simultaneous optimization of these objectives may lead to better solutions than the results achieved when both objectives are optimized separately.

The remainder of this chapter is organized as follows. Section 16.2 briefly discusses the related work on ant clustering and multiobjective clustering. Section 16.3 introduces and describes the proposed algorithm, MACC. Section 16.4 presents and discusses the results achieved, while Section 16.5 presents the conclusions and outlines future works.

16.2 Related Work

Social insects (e.g. ants, termites, bees, and wasps) distinguish themselves by their self-organization [10, 11]. The use of the social insects metaphor to solve computer problems such as combinatorial optimization, communications networks, or robotics is increasing [12]. Ant colony optimization (ACO) [13] is a class of algorithms based on social insects that has its origins in the ant foraging behavior. This behavior consists in ants depositing a chemical pheromone as they move from a food source to their nest, and foragers following such pheromone trails [12].

As mentioned, clustering approaches inspired by social insects (e.g. [4, 5, 6, 7, 8]) all optimize only a single objective. The algorithm presented in [6] relies mainly on

pheromone trails to guide ants to select a cluster for each data object, while a local search is required to randomly improve the best solution before updating pheromone trails. In this algorithm, ants visit data objects one by one in a sequence and select clusters for data objects by considering pheromone information. Pheromone deposition depends on an objective function value and on an evaporation rate.

In [4], the authors propose the Ant Colony Optimization for Clustering (ACOC) which extends the algorithm proposed in [6] by introducing the concept of dynamic cluster centers in the ant clustering process, and by considering pheromone trails and heuristic information together at each solution construction step. This heuristic information is the Euclidean distance between data objects and clusters centers of ants. It serves to guide artificial ants to group data objects into proper clusters.

The algorithm presented by Yang and Kamel in [7] employs three ant colonies to solve an important subset of the clustering problem known as the cluster ensemble problem. In this problem one needs to combine multiple clustering, formed from different aspects of the same data set, into a single unified clustering [14]. This approach is based on a hypergraph to combine clustering produced by three colonies. In each colony ants move with different speeds: constant, random, and randomly decreasing. Each ant colony projects data objects randomly onto a plane and the clustering process is done by ants picking up or dropping down objects with different probabilities. Authors used a different ant clustering model inspired by the cemetery organization behavior proposed by Deneubourg [15].

In [8] Yang and Kamel have presented an extended version of [7]. They added a *centralized* element to compute the clustering: a queen ant agent. This agent receives the results produced by all colonies, calculates a new similarity matrix and broadcasts to each ant colony of the model. Each colony re-clusters the data using the new information received.

Besides these works based on ACO for single objective clustering, we discuss next some multiobjective clustering approaches that are based on evolutionary algorithms. In [9] the authors present a multiobjective evolutionary algorithm called MOCK. This algorithm is able to simultaneously optimize two complementary objectives based on cluster connectedness and compactness. MOCK returns a set of different trade-off partitioning over a range of different numbers of clusters.

Another work is proposed by Faceli et al. [3]. This approach, called multiobjective clustering ensemble, MOCLE, is based on ideas from cluster ensembles and multiobjective clustering. Notice that the cluster ensemble is different from clustering. It is a subset of the clustering problem, which combines multiple clusterings, formed from different aspects of the same data set, into a single unified clustering. The goal is to create a single clustering that best characterizes a set of clustering, without using the original data points already used to generate the clusterings. MOCLE uses the results of several different clustering algorithms and returns a set of solutions. According to the authors these solutions are diverse (revealing the different structures of the data) and concise (may be analyzed by domain experts). Finally, we remark that a multiagent ensemble approach was also proposed [16] but it has a different purpose.

Table 16.1: Comparison between MACC and related ant and multiobjective clustering approaches

	ACOC	Shelokar et at.	Yang and Kamel	**MACC**	MOCLE	MOCK
Data selecting process	stochastic	sequential	stochastic	stochastic	-	-
Multiple colonies	no	no	yes	yes	-	-
Paradigm	ACO	ACO	Cemetery organization	ACO	evolutionary algorithms	evolutionary algorithms
Pheromone update	by elitist ants	by elitist ants	-	by all ants	-	-
Memory of visited data	yes	no	no	no	-	-
Clustering type	single objective	single objective	ensemble	multiobjective	ensemble and multiobjective	multiobjective

Table 16.1 summarizes some differences and similarities between the MACC algorithm, ACO based clustering approaches, and approaches for multiobjective clustering. The main difference between MACC and ACO approaches is that MACC performs the clustering with two colonies optimizing two objectives. MOCK and MACC optimize multiobjective functions in a different way. While MOCK uses evolutionary algorithms, MACC is inspired by social insects.

16.3 MACC – Multi Ant Colony Clustering Algorithm

Given a data set $\mathcal{D}$ containing $|\mathcal{D}|$ data objects with $|\mathcal{X}|$ attributes and a predetermined number of clusters $|\mathcal{Z}|$, the proposed algorithm has to find a clustering configuration combining the results of two objective functions. Other approaches inspired by ACO [4, 6] also use the number of clusters as a parameter for *a priori*.

MACC uses two colonies, each one with K agents working to group a data set according to two different objectives. Each colony C works in parallel over the same data set. However, each one optimizes a different objective function. In this work, we use two different objective functions: compactness (*dev*) and connectedness (*con*), using one colony for each objective. Colony C^1 optimizes the objective based on compactness of clusters. Compactness measures the overall summed distances between objects and their corresponding cluster center. It is defined by Equation 16.1, where $\mathcal{Z}$ is the set of all clusters, o is a data object, c_j is the center (or *centroid*) of cluster Z_j and $d(o,c_j)$ is the Euclidean distance (Equation 16.2) between o and c_j.

$$dev(\mathcal{Z}) = \sum_{Z_j \in \mathcal{Z}} \sum_{o \in Z_j} d(o,c_j) \tag{16.1}$$

$$d(o,c_j) = \sqrt{\sum_{i=1}^{|\mathcal{X}|} (o_i - c_{ji})^2} \tag{16.2}$$

The other colony, C^2, optimizates the connectedness. It is inspired on the Shared Nearest Neighbor clustering algorithm [17], which defines the similarity between pairs of objects in terms of how many nearest neighbors two objects share. In MACC we use this approach to group a set of data in the following way: Each object of the data set has a list $\mathcal{N}$ of neighboring data (size $|\mathcal{N}|$). An agent groups an object o according to the similarity between o and its neighbors that are in the cluster. However these neighbors are considered only if $o \in \mathcal{N}$. The connectedness reflects how frequently neighboring objects have been placed in the same group. It is computed according to Equation 16.3, where nn_{iq} is the *q-th* nearest neighbor of o. This definition of connectedness was also employed in [3, 9].

$$con(\mathcal{Z}) = \sum_{i=1}^{|\mathcal{D}|} \sum_{q=1}^{|\mathcal{N}|} a(o_i, nn_{iq}), \text{ where } a(o_i, nn_{iq}) = \begin{cases} \frac{1}{q}, \text{ if } \nexists Z_j \text{such as } o_i, nn_{iq} \in Z_j \\ 0 \quad \text{otherwise} \end{cases} \tag{16.3}$$

Table 16.2 summarizes the main parameters used by MACC. Colonies C^1 and C^2 use global pheromone matrixes G^1 and G^2 of size $|\mathcal{D}| \times |\mathcal{Z}|$ to store the pheromone values. Each agent carries a local pheromone matrix L of size $|\mathcal{D}| \times |\mathcal{Z}|$ with values of pheromone which are used to group the objects. The whole information about the global pheromone matrix G is not known by the agents. They only know their own local pheromone matrix L, which has a partial view of the global pheromone matrix. Let us denote k_i^n the *i-th* agent of colony C^n. Each k_i^1 of colony C^1 carries a list $\mathcal{M}$ of centers of size $|\mathcal{Z}|$ to store their own cluster centers and update them each T steps. Each agent k_i^2 of colony C^2 carries an object $g_i \in \mathcal{D}$ which has a list $\mathcal{N}$ of neighbors. In the clustering process each agent k_i^2 needs to update the neighbors list related to its object g_i so that this list contains data that is similar to the object g_i.

MACC is described in Algorithm 6. It starts with the initialization of the colonies and their agents. The elements of the global pheromone matrix G^1 and the local pheromone matrix L^1 are initialized with zero, indicating no pheromone. The list of centers $\mathcal{M}$ of each k_i^1 is initialized with randomly chosen data. The elements of G^2 and L^2, which calculate the connectivity, are initialized in a different way because Procedure 3, needs non-zero values for G^2 and L^2. Thus G^2 and L^2 are initialized by agents of colony C^2 according to Procedure 1. Each k_i^2 receives a randomly chosen

Table 16.2: Main parameters of the MACC algorithm

Parameter	Description	Parameter	Description
$\mathcal{D}$	data set (size $\|\mathcal{D}\|$)	$\mathcal{M}$	list of centers (size $\|\mathcal{Z}\|$)
$\mathcal{X}$	set of attributes (size $\|\mathcal{X}\|$)	ρ	persistence of the pheromone trail
$\mathcal{Z}$	set of clusters (size $\|\mathcal{Z}\|$)	T	interval size (steps) to redefine $\mathcal{M}$
C	colony	z	number of elements in cluster Z
G^n	global pheromone matrix of colony C^n (size $\|\mathcal{D}\| \times \|\mathcal{Z}\|$)	L^n	local pheromone matrix of colony C^n (size $\|\mathcal{D}\| \times \|\mathcal{Z}\|$)
$\mathcal{N}$	neighbors list (size $\|\mathcal{N}\|$)	K	number of agents in colony C^n

object g_i to carry. The list $\mathcal{N}$ of neighbors of each datum o is also initialized with $|\mathcal{N}|$ randomly chosen elements of the data set $\mathcal{D}$.

After the initialization process, agents of both colonies start to cluster the data. Colonies work separately, each one with an objective. At the beginning, each agent k randomly selects an object o to group. After that, agents in C^1 and C^2 work according to Procedure 2 and 3 respectively.

We now explain the behavior of colony C^1. Each k_i^1 needs to redefine its list $\mathcal{M}$ of centers in a distributed way. The parameter T regulates the execution of this process. Each attribute x_v of the new centroid $c_j \in \mathcal{M}$ is calculated according to Equation 16.4, where z_j is the total number of data in cluster Z_j.

$$x_v = \frac{\sum_{i=1}^{z_j} x_{vi}}{z_j}, \quad v = 1...|\mathcal{X}|, \quad j = 1...|\mathcal{Z}|, \tag{16.4}$$

All necessary information to redefine the list of centers is searched from the local pheromone matrix L^1 of the agent k_i^1. Ant k_i^1 calculates the similarity between the object o and each centroid c_j of its list of centers ($\mathcal{M}$) according to Equation 16.2. Each agent adds τ_{oj}, a pheromone concentration of object o associated to the cluster Z_j, in the global pheromone matrix G^1 . This matrix is updated according to Equation 16.5, where ρ is a constant indicating the persistence of the trail, $(0 \leq \rho \leq 1)$; $(1-\rho)$ is the evaporation rate; t is the iteration number; and τ_{oj} is calculated according to Equation 16.6 for C^1.

$$\tau_{oj}(t) = (1-\rho)\tau_{oj}(t-1) + \sum_{k=1}^{K} \tau_{oj}^k \tag{16.5}$$

$$\tau_{oj} = 1 - d(o, c_j) \tag{16.6}$$

After updating the global pheromone matrix G^1, agents of colony C^1 update their local pheromone matrix L^1. They copy the pheromone concentration τ_{oj} relative to object o and each cluster Z_j from the global pheromone matrix G^1 to the local pheromone matrix L^1.

Next we explain the behavior of colony C^2. Agents working in colony C^2 group the objects in a different way. To calculate the pheromone τ_{oj} they consider the neighborhood of the object. They first update the neighbors list of their object g_i they carry. This process is performed by each agent k_i^2 when it selects an object o to group. The agent calculates the similarity between the selected object o and the data of the list of neighbors of its object g_i according to Equation 16.2. If the object o is more similar to the object g_i than to every datum in the list $\mathcal{N}$ of neighbors, then the object o will be added to $\mathcal{N}$ replacing the less similar object.

After updating $\mathcal{N}$ each k_i^2 starts the clustering process. To group an object o in a cluster Z_j it considers the list $\mathcal{N}$ of o. Then, when k_i^2 randomly selects an object o to group, it checks in its local pheromone matrix L^2 how many neighbors of o are in each cluster Z_j and updates G^2 according to Equation 16.5. It calculates the pheromone value τ_{oj} for the object o and cluster Z_j according to Equation 16.7, where o_i indicates the object that needs to be compared with o ($o_i \in Z_j$), P is the number of objects that are neighbors of o and are in the cluster Z_j, and V is the number of objects that are not neighbors of o and are in the cluster Z_j. $\mathcal{N}_o$ and $\mathcal{N}_{o_i}$

are the list of neighbors of object o and o_i respectively, and $d(o,o_i)$ is the Euclidean distance between the object o and o_i.

$$\tau_{oj} = \begin{cases} \frac{\sum_{i=1}^{P} \delta(o)}{V} & \text{if } P < V \\ 1 & \text{otherwise,} \end{cases} \tag{16.7}$$

$$\text{where } \delta(o) = \begin{cases} d(o,o_i) & \text{if } o \in \mathcal{N}_{o_i} \quad \text{and} \quad o_i \in \mathcal{N}_o \\ 0 & \text{otherwise,} \end{cases}$$

After updating the global pheromone matrix G^2, agents of colony C^2 also update their local pheromone matrix L^2. This process is carried out in the same way as for colony C^1 by copying G^2 relative to the object o and to each cluster Z_j to L^2. At the end of the process, each object will have different values of pheromone for each group. The object will belong to the cluster with the highest amount of pheromone.

Algorithm 6: MACC algorithm

```
initialize C^1 and C^2 with K agents;
initialize 𝒩 with random objects;
initialize G^1 with pheromone value equal to 0;
initialize L^1 of each agent k_i^1 with pheromone value equal to 0;
initialize G^2 and L^2 according Procedure 1;
initialize ℳ of each agent k_i^1 with random objects;
initialize parameters T, ρ, |𝒵|;
repeat for each step
    foreach colony C^n do
        foreach agent i do
            randomly choose an object o to group;
            if agent i is working from colony C^1 then
                cluster o using Procedure 2;
            else if agent i is working from colony C^2 then
                cluster o using Procedure 3;
until numSteps;
combine both global pheromone matrices G^1 and G^2: G = G^1 + G^2;
```

Procedure 1 `initialize` G^2 and L^2

```
foreach agent k_i^2 do
    randomly choose an object o to group;
    randomly choose a cluster Z_j to put o;
    agent initializes G^2 for the object o and cluster Z_j with value 0.1;
L^2 ← G^2 ;
```

Procedure 2 `compactness`

```
1 if step % T == 0 then
2     foreach cluster Z_j, j ∈ {0,...,|Z|} do
3         foreach attribute x ∈ {0,...,|X|} do
4             calculate x according Equation 16.4;
5 foreach cluster Z_j, j ∈ {0,...,|Z|} do
6     calculate the similarity between o and c_j according to Equation 16.2;
7     update G^1 for the object o and cluster Z_j according to Equation 16.5;
8     update local pheromone matrix L^1 ;
```

Procedure 3 `connectedness`

```
1 foreach object i ∈ {0,...,P} do
2     if d(o,g) < d(g,i) then
3         N_g ← o;
4 foreach cluster Z_j, j ∈ {0,...,|Z|} do
5     update G^2 according to Equation 16.7;
6     update local pheromone matrix L^2 ;
```

16.4 Experiments and Results

We have performed experiments to investigate the quality of the proposed multiobjective ant clustering algorithm using public domain data set. The Iris data set from the Machine Learning Repository [18] was used (as in other multiobjective algorithms). The data set contains 3 classes referring to a type of Iris plant (Setosa, Versicolour and Virginica), with 4 attributes and 50 instances each.

For comparison we use the MOCK and the Yang & Kamel algorithms. Experiments were repeated 50 times (comparison with MOCK) or 10 times (with Yang & Kamel). The MACC algorithm parameter values used are: $T = 40$, $\rho = 0.3$, $|\mathcal{N}| = 20\%$ of $|\mathcal{D}|$, and $K = 3 \times |\mathcal{D}|$, and the number of iterations (*numSteps*) was 2000. These values have chosen after several tests with different values for each parameter.

To assess the performance of the clustering produced according to both objectives (compactness and connectedness) we have performed the experiments in two phases. In the first, we have investigated the quality of results of the two colonies C^1 and C^2 working independently. Each colony performed the clustering using a pheromone matrix (G^1 and G^2 respectively). In the second phase we have studied the results of both global pheromone matrix combined. This combination is performed by adding the pheromone values of both matrices. The main idea of the second phase is to optimize both objectives to find better results than when using the colonies optimizing a single objective separately.

We use the F-measure evaluation function expressed in Equation 16.8; it can take values in the interval [0,1] and should be maximized. This function is used in

[3, 7, 8] as well. The overall F-measure for the clustering is computed by Equation 16.9, where $p(i,j) = \frac{n_{ij}}{n_j}$, $r(i,j) = \frac{n_{ij}}{n_i}$, n_{ij} is the number of objects of class i within cluster j, n_j is the total number of objects within cluster j, and n_i is the number of objects of class i.

$$F(i,j) = \frac{2 \times p(i,j) \times r(i,j)}{p(i,j) + r(i,j)}, \tag{16.8}$$

$$F = \sum \frac{n_i}{D} \times max\{F(i,j)\} \tag{16.9}$$

Table 16.3 shows both phases of the clustering obtained by MACC. As we can see, the objective of MACC was achieved: the multiobjective clustering process (*combined*) works better than each colony working separately (*dev* and *con*). Table 16.4 compares MACC and MOCK while 16.5 Table shows the performance of MACC in comparison to Yang & Kamel algorithm. Entries in Table 16.4 show the sample median and interquartile range (IQR) for the F-measure while Table 16.5 depicts the average and standard deviation for the F-measure. These different measures are used to make both comparisons possible. Table 16.4 shows that MACC outperforms MOCK both when colonies work separately (the first phase) and after the combination of both pheromone matrices (the second phase).

Table 16.5 shows that our approach performs similarly to the Yang & Kamel algorithm, when the combined matrices are used. Moreover, it must be also noticed that the Yang & Kamel algorithm uses a centralized queen ant agent that receives the results produced by all colonies, computes a new similarity matrix and broadcasts to each ant colony of the model, while the agents of MACC do the clustering in a partially distributed way without knowledge about the whole environment.

Table 16.3: F-measure for MACC for the Iris data set (50 repetitions).

Measure	MACC (*dev*)	MACC (*con*)	MACC (*combined*)
Average	0.8857	0.8997	0.9215
Std. deviation	0.0017	0.1033	0.0686

Table 16.4: F-measure for MOCK and MACC for the Iris data set (50 repetitions).

Measure	MACC (*dev*)	MACC (*con*)	MACC (*combined*)	MOCK
Median	0.8853	0.9286	0.9397	0.8346
IQR	0	-0.0609	-0.0676	0.0076

Table 16.5: F-measure for Yang & Kamel and MACC for Iris data set (10 repetitions).

Measure	MACC (*dev*)	MACC (*con*)	MACC (*combined*)	Yang & Kamel
Average	0.8910	0.9219	0.9326	0.9494
Std. Deviation	0.0059	0.0407	0.0407	0.0038

16.5 Conclusion and Future Work

This chapter presented MACC, a new clustering algorithm based on ideas of multi ant colony and multiobjective clustering. The central idea of MACC is to simultaneously use several ant colonies, each optimizing one objective. Here we have performed experiments where one colony minimizes the compactness, while another maximizes the connectivity of clusters. The simultaneous optimization of these objectives leads to better solutions than those achieved when both objectives are optimized separately.

MACC was compared with a multiobjective clustering algorithm, MOCK, and with a multi ant colony, the Yang & Kamel algorithm. Results were extremely encouraging and therefore we plan to use it in other datasets, including some that are related to experiments on microarrays, where the objective is to find correlations among several genes that are expressed.

Besides, we plan to try another way to perform the multiobjective clustering: each colony collaborates with the results of each other by adding pheromone values in the corresponding global pheromone matrix. We also plan to test the optimization of conflicting objectives.

Acknowledgements This research is partially supported by the Air Force Office of Scientific Research (AFORS) (grant number FA9550-06-1-0517) and by the Brazilian National Council for Scientific and Technological Development (CNPq).

References

1. Cao, L., Luo, C., Zhang, C.: Agent-mining interaction: An emerging area. In: AIS-ADM, Springer (2007)
2. Santos, C.T., Bazzan, A.L.C.: Integrating knowledge through cooperative negotiation – a case study in bioinformatics. In Gorodetsky, V., Liu, J., Skormin, V.A., eds.: Proceedings of the International Workshop on Autonomous Intelligent Systems: Agents and Data Mining. Number 3505 in Lecture Notes in Artificial Intelligence, Springer-Verlag (2005) 277–288
3. Faceli, K., Carvalho, A.C.P.L.F., Souto, M.C.P.: Multi-objective clustering ensemble. In: Proceedings of the Sixth International Conference on Hybrid Intelligent Systems (HIS 06), Washington, DC, USA, IEEE Computer Society (2006) 51
4. Kao, Y., Cheng, K.: An ACO-based clustering algorithm. In: Proceedings of the Fifth International Workshop on Ant Colony Optimization and Swarm Intelligence - ANTS 2006. Volume 4150 of Lecture Notes in Computer Science., Brussels, Belgium, Springer (2006) 340–347
5. Lumer, E.D., Faieta, B.: Diversity and adaptation in populations of clustering ants. In: Proceedings of the third international conference on Simulation of adaptive behavior: from ani-

mals to animats, Cambridge, MA, USA, MIT Press (1994) 501–508
6. Shelokar, P.S., Jayaraman, V.K., Kulkarni, B.D.: An ant colony approach for clustering. Analytica Chimica Acta **509** (2004) 187–195
7. Yang, Y., Kamel, M.: Clustering ensemble using swarm intelligence. In: Proceedings of the Swarm Intelligence Symposium (SIS 03), Indianapolis, USA (2003) 65–71
8. Yang, Y., Kamel, M.: An aggregated clustering approach using multi-ant colonies algorithms. Pattern Recongnition **39** (2006) 1278–1289
9. Handl, J., Konwles, J.: Exploiting the trade-off - the benefits of multiple objectives in data clustering. In: Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization (EMO 2005), Springer Verlag (2005) 547–560
10. Camazine, S., Deneubourg, J.D., Franks, N.R., Sneyd, J., Theraulaz, G., Bonabeau, E.: Self-Organization in Biological Systems. Princeton University Press, Princeton, N.J. (2003)
11. Gordon, D.: The organization of work in social insect colonies. Nature **380** (1996) 121–124
12. Bonabeau, E., Theraulaz, G., Dorigo, M.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, New York, USA (1999)
13. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man and Cybernetics Part B: Cybernetics **26** (1996) 29–41
14. Strehl, A., Ghosh, J.: Cluster ensembles: a knowledge reuse framework for combining partitionings. In: Proceedings of the Eighteenth National Conference Intelligence, Menlo Park, CA, USA, American Association for Artificial Intelligence (2002) 93–98
15. Deneubourg, J.L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., Chrétien, L.: 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, Canbridge, MA, USA, MIT Press (1991) 356–363
16. Agogino, A., Tumer, K.: Efficient agent-based cluster ensembles. In Stone, P., Weiss, G., eds.: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, AAMAS '06, New York, NY, USA, ACM (2006) 1079–1086
17. Ertöz, L., Steinbach, M., Kumar, V.: A new shared nearest neighbor clustering algorithm and its applications. In: Proceedings of the International Conference on Data Mining (2nd SIAM). Volume 4., VA, IEEE Press (2002) 2642–2647
18. Asuncion, A., Newman, D.J.: UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences (2007) http://www.ics.uci.edu/∼mlearn/MLRepository.html.