A Queen Lead Ant-Based Algorithm for Database Clustering

A Master’s Paper in Computer Science
by
Jayanthi Pathapati

©2004 Jayanthi Pathapati

Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science

December 2004
Abstract

In this paper we introduce a novel ant-based clustering algorithm to solve the unsupervised data clustering problem. The main inspiration behind ant-based algorithms is the chemical recognition system of ants. In our current algorithm, we introduce a new method of ant clustering in which the queen of the ant colony controls the colony formation. We compared our results to the k-means method and to the AntClust algorithm on artificial and real data sets.
# Table of Contents

Abstract \hspace{1cm} i
Acknowledgement \hspace{1cm} iii
List of Figures \hspace{1cm} iv
List of Tables \hspace{1cm} v

1 Introduction \hspace{1cm} 1

2 Database Clustering \hspace{1cm} 1

3 Algorithm \hspace{1cm} 3
   3.1 Initialization \hspace{1cm} 5
   3.2 Training \hspace{1cm} 6
   3.3 Clustering \hspace{1cm} 7
   3.4 Final Clustering \hspace{1cm} 13

4 Performance Analysis \hspace{1cm} 13

5 Conclusion \hspace{1cm} 17

References \hspace{1cm} 17
Acknowledgements

I am very grateful to Dr. T. Bui for his constant support and guidance throughout the period of this project. It has been a great pleasure and inspiration working with him. I would also like to thank the committee members: Dr. S. Chang, Dr. Q. Ding, Dr. P. Naumov, and Dr. L. Null for reviewing this paper. Finally I would like to thank my husband Dr. S.K. Daya for his support throughout this course.
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>QLA algorithm for database clustering</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>Initialization Algorithm</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>Algorithm for Training Stage</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>Cluster Formation Algorithm</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>Rules followed when the ants meet</td>
<td>11</td>
</tr>
<tr>
<td>6</td>
<td>Local Optimization Algorithm</td>
<td>12</td>
</tr>
<tr>
<td>7</td>
<td>Algorithm For Assigning Free Ants</td>
<td>13</td>
</tr>
</tbody>
</table>
List of Tables

1  QLA-Results ........................................... 16
2  Results of all the algorithms .................... 16
1 Introduction

Database clustering is one of the fundamental operations of data mining. The main idea behind it is to identify homogeneous groups of data, based on their attributes. Obtaining the closest partition possible to the natural partition is a complex task because there is no a-priori information regarding the structure and the original classification of the data.

Artificial ants have been applied to problems of database clustering. The initial pioneering work in this field was performed by Deneunbourg[1] and Lumer and Faieta[8]. Since the early work, many authors have used artificial ants and proposed several data clustering techniques[9,10,11]. Ant based clustering algorithms seek to model the collective behavior of the real ants by utilizing the colonial odor to form clusters.

Our current paper proposes a new ant-based clustering algorithm. This algorithm incorporates several ant clustering features as well as local optimization techniques. Unlike the other ant clustering algorithms where the ants form the nests on their own, we introduce a queen for each colony to take care of the colony creation process. This algorithm does not require the number of expected clusters as part of the input to find the final partition.

The rest of the paper is organized as follows. Section 2 outlines the database clustering problem and its applications. Section 3 describes the algorithm in detail. Section 4 contains the performance analysis and the experimental results comparing our algorithm against other known algorithms. The conclusion is given in Section 5.

2 Database Clustering

The aim of database clustering is to partition a heterogeneous data set into groups of more homogenous characteristics. The formation of clusters is based on the principle of maximizing similarity between patterns in the same cluster and simultaneously minimizing the similarity between patterns belonging to distinct clusters. The Database Clustering problem could be
formally defined as follows.

*Given a database of \( n \) objects, find the best partition of the data objects into groups such that each group contains objects which are similar to each other.*

Several algorithms have been proposed for the clustering problem in published literature. Some of them are: BIRCH[14], ISODATA[2,7], CLARANS[13], P-CLUSTER[6] and DBSCAN[3].

Most of the clustering algorithms belong to two main categories: hierarchical and partitional. In hierarchical clustering algorithms, there are two main categories: Agglomerative and Divisive. In the Agglomerative approach, each object is assigned to an individual cluster and then similar clusters are merged to form larger clusters. Repetition of this process leads to the desired partition. On the other hand, Divisive algorithms begin with all the data objects placed in one cluster and then split them into smaller clusters until the final partition is obtained.

In partitional clustering algorithms, given a set of data objects and a partition criterion the objects are partitioned into clusters based on the similarities of the objects. Furthermore, clustering is performed to meet the specified criterion, such as minimization of the sum of squared distance from the means within each cluster.

Fuzzy clustering and the \( K \)-means clustering are two well known clustering techniques. In fuzzy clustering, an object can belong to many clusters. Each object is associated with a cluster membership that represents the likeliness of the object being in that cluster.

The \( K \)-means algorithm starts by choosing \( K \) cluster centers at random. It then assigns each object to the closest cluster possible and compute that cluster’s center. This step is repeated until the cluster’s centers do not change, thus obtaining the desired partition.

Association between data attributes in a given database can be found by utilizing techniques of data clustering. This plays a major role in the data analysis procedures. Database clustering has been applied in various fields including biology, psychology, commerce, and information technology. The
main areas where it has been prominently used are: image segmentation[13], statistical data analysis[4], and data compression[14] among others.

3 Algorithm

As we mentioned earlier, the main aim of the Queen Lead ant-based (QLA) algorithm is to partition the given data set into clusters of similar objects. The idea of the algorithm is to simulate the way real ants form the colonies, which are queen dominant. The ants that possess similar cuticle odor to that of the queen of a colony will join that colony. Each object of the database is treated as an ant and it follows a common set of rules[Figure 5], which leads to the required partition. A local optimization technique is applied to assist the ants in achieving good partition.

The input for the algorithm is just the database to be clustered. It does not need any other additional parameters such as the number of desired clusters or any information about the structure of the database. Each object in the database has a certain number of attributes, and attributes can be of either numeric or symbolic types. The QLA algorithm, given in Figure 1, consists of four main stages. The first stage is the initialization stage in which all the ants are initialized to the attributes of the object they represent. The second stage is the training stage in which the ants learn about their possible colony mates by meeting and learning about other ants.

The third stage consists of two phases: cluster formation and the local optimization. In the cluster formation phase, we stimulate meetings between ants so that the ants can decide which colony they belong to, thus giving us the clusters of similar ants. In the second phase, the local optimization step is designed to eliminate any ants that have been prematurely added to a colony even though they should not belong to it. It also checks to see if the queen represents a colony better than any other ant in that colony.

The last stage is the final clustering stage in which the free ants, i.e., the ants which do not belong to any colony, are added to the closest colony possible. We describe all the stages of the algorithm in greater detail in the
QLA Algorithm

Input: Database to be clustered

Stage 1: Initialization
Assign one ant for each object in the database

Stage 2: Training
Train the ants so that they learn about other ants

Stage 3: Clustering Process
Colony formation process and local optimization

Stage 4: Final Clustering
Assign all ants which do not belong to any colony to the closest colony available

return the clusters formed

Figure 1: QLA algorithm for database clustering
3 ALGORITHM

Initialization

for each ant do

    genome = the index of the object it represents

    colony = -1

    Assign all of its attribute values with the values of the object it represents

end-for

Figure 2: Initialization Algorithm

following section.

3.1 Initialization

The algorithm starts by assigning each artificial ant to each of the objects in the given input database, and then assigns all the ants with the attribute values of the object it represents. Each ant contains the following parameters.

    The *genome* parameter of the ant corresponds to an object of the data set. It is initialized to the index of the object that the ant represents. Each ant represents the same object during the whole process of clustering, thus the genome of an ant remains unchanged to the end of the algorithm.

    The *colony* parameter indicates the group to which the ant belongs and is simply coded by a number. This attribute is assigned the value $-1$ because the ant initially doesn’t belong to any colony, i.e., all the ants at the beginning of the algorithm are free ants.

    The initialization stage of the algorithm is accomplished as shown in Figure 2.
3.2 Training

The next step in the algorithm is to train the ants so that they learn about the genetic odors of other ants that are participating in the clustering. The evaluation of the similarities of the genetic odors of the ants leads to the threshold value. The threshold value is defined more formally later on. The training is done by selecting two ants randomly and stimulating a meeting between them. We have decided to do this for $\delta$ number of iterations where $\delta$ is $n_{ANTS}$ (total number of ants). When the ants meet, we find the similarity between them and update their threshold values accordingly. We also increment the age of both of the ants by 1. This is done as shown in Figure 3. The similarity and the threshold values of an ant are defined as follows.

The similarity between two ants, $a_i$ and $a_j$, denoted by $Sim(a_i, a_j)$, is calculated by using the following equation:

$$Sim(a_i, a_j) = \frac{1}{T} \sum_{k=1}^{T} Sim_k(a_i, a_j)$$

where $T$ is the number of different data types such as numeric, symbolic, etc, used to describe the objects $a_i$ and $a_j$, and $Sim_k(a_i, a_j)$ is the function that computes the similarity between the two objects of all the attributes of type $k$. The following function has been used to compute $Sim_k(a_i, a_j)$:

$$Sim_k(a_i, a_j) = 1 - \left( \frac{1}{N_k} \sum_{x=1}^{N_k} \Delta_{k_x}(a_i, a_j) \right)$$

In the above equation, $\Delta_{k_x}(a_i, a_j)$ is the function used to find the dissimilarity between the attribute $x$ of objects $a_i$ and $a_j$ of type $k$, and $N_k$ is the number of attributes of data type $k$ that represent the objects. If the data type is symbolic, then $\Delta_{k_x}(a_i, a_j)$ is 1 if the attributes $a_i$ and $a_j$ are different and 0 if they are the same. For numeric data type attributes, if the maximum and minimum values of the attribute are equal, then $\Delta_{k_x}$ is 0; otherwise it is the difference between the two attributes over the range of the attribute in the given database.

The threshold is an acceptance threshold that has a real value between 0
and 1. This threshold value is used to determine how well the ants accept each other. We have defined the threshold value of an ant to fall between the mean and the maximum similarity values of an ant. Every time an ant meets another ant in this period, we find the similarities between the two ants and update the respective mean and maximum similarities of the ants. So every time an ant has a meeting, its threshold value changes. The following equation shows how this threshold is learned:

\[ T_{a_i} \leftarrow \frac{\text{MeanSim}(a_i) + \text{MaxSim}(a_i)}{2}. \]

The \( \text{MeanSim}(a_i) \) in the above equation is the mean similarity of the ant \( a_i \) and is calculated as follows:

\[ \text{MeanSim}(a_i) = \frac{1}{|N_i|} \sum_{a_j \in N_i} \text{Sim}(a_i, a_j) \]

where \( N_i \) is the set of ants that \( a_i \) has met so far. The \( \text{MaxSim}(a_i) \) is the maximum similarity of the ant \( a_i \) and is calculated as follows:

\[ \text{MaxSim}(a_i) = \max\{\text{Sim}(a_i, a_j)\} \]

where \( a_j \) is any ant with which ant \( a_i \) had a meeting with.

### 3.3 Clustering

There are two phases in this stage. The first phase is the cluster formation phase in which clusters are created. During this phase, meetings between random ants are stimulated and the behavior between the ants is described in Figure 5. After every \( \alpha (6 \times n_{\text{ANTS}}) \) number of iterations in this step, we run a local optimization algorithm on the clusters we have found so far. This stage is performed until at least \( \beta \) number of ants join their colonies, or for \( \gamma \) times, where \( \beta \) is 50% and \( \gamma \) is \( 40 \times n_{\text{ANTS}} \). This is done as shown in Figure 4. Tests were performed with different values of \( \alpha \), \( \beta \) and \( \gamma \). For many data sets, best partitions were achieved with the above given values of \( \alpha \), \( \beta \), and \( \gamma \).
Training Stage

for $i = 0$ to $\delta$
    Select 2 ants randomly
    Find the similarity between them and
    Update their threshold and the similarity values
    Increment their ages by 1
end-for

Figure 3: Algorithm for Training Stage

When two ants meet in this stage, the threshold value is used to make the decision whether two ants accept each other or not. The $Acceptance(a_i, a_j)$ for ants $a_i$ and $a_j$ is defined as follows:

$$A(a_i, a_j) \equiv (Sim(a_i, a_j) > 0.96 \times T_{a_i}) \land (Sim(a_i, a_j) > 0.96 \times T_{a_j})$$

The $Acceptance(a_i, queen_j)$ for ant $a_i$ and queen $queen_j$ is defined as follows.

$$A(a_i, queen_j) \equiv (Sim(a_i, queen_j) > 0.98 \times T_{a_i})$$
$$\land (Sim(a_i, queen_j) > 0.98 \times T_{queen_j})$$

The $Acceptance(queen_i, queen_j)$ for queens $queen_i$ and $queen_j$ is defined as follows.

$$A(queen_i, queen_j) \equiv (Sim(queen_i, queen_j) > 0.94 \times T_{queen_i})$$
$$\land (Sim(queen_i, queen_j) > 0.94 \times T_{queen_j})$$
The \textit{Acceptance}(a_i, a_j) is defined differently between the ants and the queens. When two ants or two queens meet, they mutually accept each other if the similarity between them is little less or greater than their threshold values. However when an ant meets a queen of a colony, both similarity values should be almost close to or greater than their threshold values.

The threshold and the similarity values are calculated using the same formulas as shown in the training stage. The \textit{MeanSim}(a_i) and \textit{MaxSim}(a_i) of the ant \(a_i\) are calculated using different formulas because as the ants get older, they update their threshold values less frequently, and the chances that they move from one colony to another is quite less. The following equations are used to update \(a_i\)'s mean and maximum similarities when it meets ant \(a_j\):

\[
\text{MeanSim}(a_i) = \frac{\text{Age}_{a_i}}{\text{Age}_{a_i} + 1} \text{MeanSim}(a_i) + \frac{1}{\text{Age}_{a_i} + 1} \text{Sim}(a_i, a_j)
\]

\[
\text{MaxSim}(a_i) = 0.8 \times \text{MaxSim}(a_i) + 0.2 \times \text{Sim}(a_i, a_j)
\]

only if

\[
\text{Sim}(a_i, a_j) > \text{MaxSim}(a_i)
\]

The next phase in this stage is the Local Optimization Phase. The Local Optimization algorithm is described in Figure 6. The aim in doing local optimization is to make sure that the queen of a colony is the best representative of the colony and to remove the ants that should not belong to that colony. The reasoning behind the need for the queen being the best representative of the colony is that crucial decisions, like addition and deletion of ants, are made by the queen. It also allows us to have a better partition without having to have more meetings between an ant and its potential colony mates.

At the end of this stage, we have most of the ants in their respective colonies. We check if any two queens are similar to each other and merge them if they accept. In the next stage, we address the clustering of the remaining ants.
Cluster Formation

Until 50% of the total ants find their colonies or
for i = 0 to $\gamma$ do
  Randomly select 2 ants
  Stimulate a meeting between them
  Follow the rules in the meeting algorithm and
take appropriate action

  if $\alpha$ is 0
    Run the local optimization algorithm
  end-if
end-for or
end-until

for each colony $C$ do
  Check if the queen of colony $C$ is similar to any
  other colony’s queen and merge them if they accept
end-for

Figure 4: Cluster Formation Algorithm
**Figure 5:** Rules followed when the ants meet

Rules when two ants $X$ and $Y$ meet

**Case 1:** If both of the ants do not belong to any colony and if they accept each other
- Create a new colony and add these two ants to that colony and make one of them the queen of that colony

**Case 2:** If one of the ants belongs to a colony A and the other is free
- If they accept each other and the queen of the colony A accepts the free ant
  - Add the free ant to the colony A and update the queen of that colony with the information that the new ant is being added

**Case 3:** If both ants are present in the same colony A
- If they accept each other
  - leave them as they are
- Else (they do not accept each other)
  - Check if the queen of the colony A accepts the ants
  - Delete the ants that the queen has not accepted and inform the queen about the removal of the ants

**Case 4:** If the ants are present in different colonies A and B
- If the ants accept each other and the queens of the colonies A and B accept each other
  - Merge both colonies into one colony and make the queen of the bigger colony the queen of the new colony formed
Local Optimization
for each colony do
   find the ant \( a \) that is most similar to all of
   the rest of the ants in that colony. This is done in
   a two step process as follows:
   Step 1:
   for each ant \( a \) in the nest do
      Find the similarities between the ant \( a \)
      and all of other ants in the colony. Calculate the
      average of all those similarities to obtain the
      average similarity of ant \( a \)
   end-for
   Step 2:
   Pick the ant \( a \) with the highest average similarity
   in the colony

   If ant \( a \) is not the present queen of that
   colony then make \( a \) the queen

   Find the average similarity of the colony and remove the
   ants that has the average similarity less than
   \( 0.97 \times CS \), where CS is the colony’s average similarity
end-for

Figure 6: Local Optimization Algorithm
Final Clustering

for each \textit{free ant} do

Find the similarities between the \textit{free ant} and all the
queens of the available colonies. Then find the most
similar queen and assign the \textit{free ant} to the colony
to which that queen belongs

end-for

for each colony \textit{C} do

Check if the queen of colony \textit{C} is similar to any
other colony’s queen and merge them if they accept

end-for

\textbf{Figure 7:} Algorithm For Assigning Free Ants

3.4 Final Clustering

There might be some free ants that have not yet joined any colony. For each
of the free ants, we find the closest colony possible and add the ant to that
colony. The algorithm for assigning free ants is described in Figure 7.

4 Performance Analysis

The algorithm was tested on some artificially generated databases and some
real databases from the Machine Learning Repository[15]. These databases
are used as a benchmark as they have been used to test a number of different
ant-based clustering algorithms. Thus the results can be compared to other
algorithms. The databases that we used have up to 1473 objects and 35 attributes. The artificial data sets Art1, Art2, Art3, Art4, Art5 and Art6, are generated according to Gaussian or uniform laws in such a way that each poses a unique situation for the algorithm to handle in the process of clustering.

Generally the real data sets are more difficult to cluster because they may be noisier than the artificial data sets. The real data sets that we used to test our algorithm are Soybean, Iris, wine, Thyroid, Glass, Haberman, Ecoli, Pima and CMC.

The Soybean data set has the data of four kinds of soybean diseases. The data set contains 45 instances, each with 35 attributes. This data set gives the ability to test the algorithm on a data set with many attributes.

The Iris data set contains 3 classes of 50 instances each, where each class refers to a type of Iris plant. The wine data set is the result of an analysis of wines grown at a particular place. It contains 178 instances in 3 classes. The Thyroid data set contains 214 instances of diagnosis of thyroid. It has three classes each representing a kind of a thyroid disorder (like hyperthyroid and hypothyroid).

The Glass data set is obtained from the study of classifications of types of glass. This data set contains 9 classes, hence it is used to test the algorithm for the data set that has records from many classes. The Haberman data set is obtained from a study on survival of patients who had undergone surgery for breast cancer. It has 306 instances with 2 classes in it.

The Ecoli data set contains information about localization sites of protein. It has 8 classes and 336 instances in it. The Balance-Scale data set was generated to model psychological experimental results. It has 625 instances in 3 classes. The Pima data set has 798 instances of the diabetes diagnosis for the Pima Indian heritage women, the data set has 2 classes in it. The CMC data set was obtained from 1987 National Indonesia Contraceptive Prevalence Survey. It has 1473 instances in 3 classes.

To test the performance of our algorithm, we have used the classification success $C_s$ measure developed by Fowlkes and Mallows[5] and modified by
This measure allows us to compare the clusters obtained by our algorithm to the real clusters in the database. We denote the number of objects in the database by \( N \). The following equation has been used to find the clustering success \( C_s \):

\[
C_s = 1 - \frac{2}{N(N-1)} \sum_{i,j \in 1..N^2, i < j} \epsilon_{ij}
\]

where \( \epsilon_{ij} \) is 0 if the ants (\( i \) and \( j \)) are either in the same colony or in different colonies as they are in the expected solution; otherwise \( \epsilon_{ij} \) is 1.

This algorithm is implemented in C++ and run on a PC with a Pentium IV 2.4GHz processor and 512MB of RAM. For each data set the algorithm is run 50 times. The mean clustering success, mean number of clusters, and their respective standard deviations are computed for each data set.

Table 1 shows the real number of clusters, attributes, mean clustering success [standard deviation], mean number of clusters [standard deviation], and the average running time of the QLA algorithm for selected data sets. It is clear from Table 1 that the QLA algorithm works well on both the small and large databases. In Table 1:

- \( O \) represents the number of objects;
- \( A \) represents the number of attributes;
- \( N \) represents the number of clusters in the real data;
- \( \text{MeanCS} \) represents the mean clustering success;
- \( \text{MeanNC} \) represents the mean number of clusters;
- \( \text{Std} \) represents the standard deviation;
- \( \text{AvgRT} \) represents the average running time in seconds.

For most of the data sets, the QLA algorithm managed to achieve the clustering success rate close to the best clustering success rate or a comparable one. The results of QLA, AntCLust, and K-means algorithms for some data sets are shown in Table 2.
Table 1: QLA-Results

<table>
<thead>
<tr>
<th>Database</th>
<th>O</th>
<th>A</th>
<th>N</th>
<th>MeanCS[Std]</th>
<th>MeanNC[Std]</th>
<th>AvgRT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Soybean</td>
<td>47</td>
<td>35</td>
<td>4</td>
<td>0.92[0.03]</td>
<td>5.84[0.70]</td>
<td>0.02</td>
</tr>
<tr>
<td>Iris</td>
<td>150</td>
<td>4</td>
<td>3</td>
<td>0.84[0.02]</td>
<td>4.98[1.00]</td>
<td>0.22</td>
</tr>
<tr>
<td>Wine</td>
<td>178</td>
<td>13</td>
<td>3</td>
<td>0.81[0.12]</td>
<td>4.56[1.2]</td>
<td>0.26</td>
</tr>
<tr>
<td>Thyroid</td>
<td>215</td>
<td>5</td>
<td>3</td>
<td>0.80[0.07]</td>
<td>4.92[1.26]</td>
<td>4.87</td>
</tr>
<tr>
<td>Glass</td>
<td>214</td>
<td>9</td>
<td>7</td>
<td>0.65[0.03]</td>
<td>6.62[1.61]</td>
<td>5.39</td>
</tr>
<tr>
<td>Haberman</td>
<td>306</td>
<td>3</td>
<td>2</td>
<td>0.94[0.01]</td>
<td>3.24[0.90]</td>
<td>0.63</td>
</tr>
<tr>
<td>Ecoli</td>
<td>336</td>
<td>8</td>
<td>8</td>
<td>0.79[0.06]</td>
<td>7.74[2.12]</td>
<td>4.80</td>
</tr>
<tr>
<td>Balance-Scale</td>
<td>625</td>
<td>4</td>
<td>3</td>
<td>0.60[0.01]</td>
<td>11.09[1.23]</td>
<td>0.30</td>
</tr>
<tr>
<td>Pima</td>
<td>798</td>
<td>8</td>
<td>2</td>
<td>0.54[0.01]</td>
<td>3.34[1.30]</td>
<td>0.90</td>
</tr>
<tr>
<td>CMC</td>
<td>1473</td>
<td>9</td>
<td>3</td>
<td>0.61[0.00]</td>
<td>15.16[2.04]</td>
<td>0.77</td>
</tr>
<tr>
<td>Art1</td>
<td>400</td>
<td>2</td>
<td>4</td>
<td>0.84[0.01]</td>
<td>7.46[1.15]</td>
<td>0.71</td>
</tr>
<tr>
<td>Art2</td>
<td>1000</td>
<td>2</td>
<td>2</td>
<td>0.78[0.04]</td>
<td>5.9[1.66]</td>
<td>34.28</td>
</tr>
<tr>
<td>Art3</td>
<td>1100</td>
<td>2</td>
<td>4</td>
<td>0.80[0.03]</td>
<td>6.9[1.72]</td>
<td>40.20</td>
</tr>
<tr>
<td>Art4</td>
<td>200</td>
<td>2</td>
<td>2</td>
<td>0.74[0.02]</td>
<td>6.04[1.32]</td>
<td>0.24</td>
</tr>
<tr>
<td>Art5</td>
<td>900</td>
<td>2</td>
<td>9</td>
<td>0.85[0.02]</td>
<td>8.54[1.25]</td>
<td>2.77</td>
</tr>
<tr>
<td>Art6</td>
<td>400</td>
<td>8</td>
<td>8</td>
<td>0.95[0.02]</td>
<td>6.52[1.40]</td>
<td>6.41</td>
</tr>
</tbody>
</table>

Table 2: Results of all the algorithms

<table>
<thead>
<tr>
<th>Database</th>
<th>O</th>
<th>A</th>
<th>N</th>
<th>QLA [MeanCS[Std]]</th>
<th>AntClust [MeanCS[Std]]</th>
<th>K-means [MeanCS[Std]]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Soybean</td>
<td>47</td>
<td>35</td>
<td>4</td>
<td>0.92[0.03]</td>
<td>0.93[0.04]</td>
<td>0.91[0.08]</td>
</tr>
<tr>
<td>Iris</td>
<td>150</td>
<td>4</td>
<td>3</td>
<td>0.84[0.02]</td>
<td>0.78[0.01]</td>
<td>0.86[0.03]</td>
</tr>
<tr>
<td>Thyroid</td>
<td>215</td>
<td>5</td>
<td>3</td>
<td>0.80[0.07]</td>
<td>0.84[0.03]</td>
<td>0.82[0.00]</td>
</tr>
<tr>
<td>Glass</td>
<td>214</td>
<td>9</td>
<td>7</td>
<td>0.65[0.03]</td>
<td>0.64[0.02]</td>
<td>0.68[0.01]</td>
</tr>
<tr>
<td>Pima</td>
<td>798</td>
<td>8</td>
<td>2</td>
<td>0.54[0.01]</td>
<td>0.54[0.01]</td>
<td>0.56[0.00]</td>
</tr>
<tr>
<td>Art1</td>
<td>400</td>
<td>2</td>
<td>4</td>
<td>0.84[0.01]</td>
<td>0.78[0.03]</td>
<td>0.89[0.00]</td>
</tr>
<tr>
<td>Art2</td>
<td>1000</td>
<td>2</td>
<td>2</td>
<td>0.78[0.04]</td>
<td>0.93[0.02]</td>
<td>0.96[0.00]</td>
</tr>
<tr>
<td>Art3</td>
<td>1100</td>
<td>2</td>
<td>4</td>
<td>0.80[0.03]</td>
<td>0.85[0.02]</td>
<td>0.78[0.02]</td>
</tr>
<tr>
<td>Art4</td>
<td>200</td>
<td>2</td>
<td>2</td>
<td>0.74[0.02]</td>
<td>0.77[0.05]</td>
<td>1.00[0.00]</td>
</tr>
<tr>
<td>Art5</td>
<td>900</td>
<td>2</td>
<td>9</td>
<td>0.85[0.02]</td>
<td>0.74[0.02]</td>
<td>0.91[0.02]</td>
</tr>
<tr>
<td>Art6</td>
<td>400</td>
<td>8</td>
<td>8</td>
<td>0.95[0.02]</td>
<td>0.95[0.01]</td>
<td>0.99[0.04]</td>
</tr>
</tbody>
</table>
5 Conclusion

This paper proposed a new ant-based clustering algorithm. During the process of clustering, the queen of the colony plays an important role in its construction because it better represents the colony than its colony mates. This method, coupled with local optimization, results in a better partition of the objects. The results obtained by this algorithm were close to the best known results for the set of benchmark databases.

Future work could be done on clustering the given data based on some given criteria. Another idea is to make the problem distributed by dividing the data set into subsets and find the solution for all the subsets in parallel. Then from the sub solutions, we could try to find the solution for the whole data set. This could make the algorithm even more competitive.

References


