Ant Colony Optimization and Data Mining

Chapter *in* Studies in Computational Intelligence · June 2011

DOI: 10.1007/978-3-642-20344-2_2

4 authors:

Ioannis Michelakos
University of Central Greece

Nikolaos Mallios

Elpiniki I Papageorgiou
University of Thessaly

Michael Vassilakopoulos
University of Thessaly

Chapter 2
Ant Colony Optimization and Data Mining

Ioannis Michelakos, Nikolaos Mallios,
Elpiniki Papageorgiou, and Michael Vassilakopoulos

Abstract. The Ant Colony Optimization (ACO) technique was inspired by the ants' behavior throughout their exploration for food. In nature, ants wander randomly, seeking for food. After succeeding, they return to their nest. During their move, they lay down pheromone that forms an evaporating chemical path. Other ants that locate this trail, follow it and reinforce it, since they also lay down pheromone. As a result, shorter paths to food have more pheromone and are more likely to be followed. ACO algorithms are probabilistic techniques for solving computational problems that are based in finding as good as possible paths through graphs by imitating the ants' search for food. The use of such techniques has been very successful for several problems. Besides, Data Mining (DM), a discipline that consists of techniques for discovering previously unknown, valid patterns and relationships in large data sets, has emerged as an important technology with numerous practical applications, due to wide availability of a vast amount of data. The collaborative use of ACO and DM (the use of ACO algorithms for DM tasks) is a very promising direction. In this chapter, we review ACO, DM, Classification and Clustering (two of the most popular DM tasks) and focus on the use of ACO for Classification and Clustering. Moreover, we briefly present related applications and examples and outline possible future trends of this promising collaborative use of techniques.

1 Introduction

The use of various optimization techniques has evolved over the years and a variety of methods have been proposed in order to approach the optimal solution, or a set of approximate solutions to a range of problems in specific areas.

Ioannis Michelakos · Michael Vassilakopoulos
2-4 Papasiopoulou st., 35100 Lamia, Greece
Dept. of Computer Science & Biomedical Informatics, University of Central Greece
e-mail: {imichelakos,mvasilako}@ucg.gr

Nikolaos Mallios · Elpiniki Papageorgiou
3rd Km Old National Road Lamia-Athens, 35100 Lamia, Greece
Dept. of Informatics and Computer Technology, Technological Educational Institute of Lamia, Greece
e-mail: {nmallios,epapageorgiou}@teilam.gr

N. Bessis, F. Xhafa (Eds.): Next Generation Data Technologies for CCI, SCI 352, pp. 31–60.
springerlink.com

Social insects like ants, perform a series of tasks as a group rather than atomically. Such behavior illustrates a high rate of swarm intelligence and classifies ants as collaborative agents. The Ant Colony Optimization (ACO) technique was introduced in the early 1990's by Marc Dorigo in his PhD Thesis [11] and was mainly inspired by the ants' behavior throughout their exploration for food. The introduction of the ACO technique [11] was followed by a number of research efforts that aimed at exploiting the behavior of ants' throughout their exploration for food in scientific problems. Computational models which apply the swarm behavior in various application areas such as finding the optimal routes (e.g. TSP problem) [13], solving hard combinatorial optimization problems (e.g. MAX-MIN Ant System) [51], biomedical data processing and classification [4], even character recognition [45] and many others have been presented.

Moreover, Data Mining (DM), a discipline that consists of techniques for discovering previously unknown[31], valid patterns and relationships in large data sets, has been acknowledged as a key research field and has emerged as an important technology with numerous practical applications, due to the wide availability of a vast amount of data. Large-scale organizations apply various DM techniques on their data, to extract useful information and patterns [24].

The objective of this survey chapter is to briefly present these two emerging technologies and outline the various ways that these technologies could be combined. The enabling technology which is derived from the collaborative use of ACO and DM (that has been rather recently proposed, for example in [41,42]) leads to improved algorithms and techniques with numerous usages in real problems and can be employed in next generation applications.

This chapter is organized as follows: Primarily, a short review of background material and state-of-the-art research on ACO, DM and their collaborative use is presented, in order to give the reader an overview of the area. Afterwards, in the next two sections, both technologies (DM and ACO) are outlined, focusing on the aspects that led to the collaboration of DM techniques with the ACO technique. Emphasis will be given in the two main ways in which ACO and DM are combined: data classification methods based on ACO [19,57] and ACO for data clustering [6,25] which are thoroughly presented in section 5. Finally, a description of a number of applications and examples where the collaborative use of ACO and DM contributes in various research areas e.g. Health, Marketing, Finance, Molecular Biology is given in section 6. Conclusions and possible future trends of research in this area follow.

2 State-of-the-Art

This section presents, in brief, background material and state-of-the-art research on ACO, DM and their collaborative use. Selected methods are presented in more detail, in the rest of the chapter.

2.1 Ant Colony Optimization State-of-the-Art

The original idea for ACO comes from observing the search of ants for food. Ants individually have limited cognitive abilities, but collectively are able to find the shortest path between a food source and their nest. In nature, ants wander randomly, seeking for food. After succeeding, they return to their nest. During their move, they lay down pheromone that forms an evaporating chemical path. Other ants that locate this trail, follow it and reinforce it, since they also lay down pheromone. As a result, shorter paths to food have more pheromone and are more likely to be followed. Thus, this positive feedback eventually leads all the ants following a single path. ACO algorithms are probabilistic techniques for solving computational problems that are based in finding as good as possible paths through graphs by imitating the ants' search for food [12,36].

ACO algorithms are inspired by the pheromone trail laying and the following behavior of some ant species, a behavior that was shown to allow real ant colonies to find shortest paths between their colony and food sources. Considering many aspects of the real ants behavior, mostly their indirect communication through pheromone trails, ACO has attracted a large number of researchers, e.g. [10,12]. During the first few years of ACO research, the focus was mainly on algorithmic advancements, trying to make ACO algorithms competitive with established metaheuristic techniques. Currently, the majority of the contributions concern, on one hand successful applications of ACO algorithms to a variety of challenging problems, while on the other hand algorithmic developments and theoretical studies for difficult optimization problems.

In the evolving area of bioinformatics a number of interesting contributions have been made. Among those, in [38] a novel ACO algorithm for the problem of predicting protein functions using the Gene Ontology (GO) structure was presented. Moreover, in [35] ACO was applied to the well-known bioinformatics problem of aligning several protein sequences. Furthermore, a number of papers, e.g. [48,49], have appeared concerning the two dimensional hydrophobic-polar (2D HP) protein folding problem. This problem is one of the most prominent problems in computational biology and with the aim of an appropriate ACO algorithm is successfully addressed.

A number of ACO algorithms has also been applied in industry in order to optimize every day's industrial problems. An indicative work is the one by Corry and Kozan [9], that tries to generate solid and better solutions in optimizing the trade-off between material handling and rearrangement costs under certain environments. Another interesting approach includes the use of ACO in scheduling cars along a line, while at the same time, satisfying capacity constraints. This car sequencing problem was described in [50] with the aim of using two different pheromone structures for the algorithm.

Additionally, a number of approaches concerning dynamic (respectively, stochastic) problems have been presented. A few, very recently proposed, ACO algorithms are presented here for the Traveling Salesman Problem (TSP) . In [29] Lopez and Blum dealt with the TSP problem with time window, which arises often in logistics. In their attempt a hybrid Beam-ACO algorithm (ACO and beam search combined) is proposed in order to minimize the travel-cost. Moreover a generalized TSP algorithm (GTSP) was presented in [59], extending the classical TSP problem. The algorithm introduces a mutation process and a local searching technique which turn to be effective.

Borkar and Das [2] introduced an ACO variant that is closer to real ants' behavior than most state-of-the-art ACO algorithms. Their algorithm uses no external supervision and the pheromone update mechanism is based only on differential path length.

Neumann, Sudholt, and Witt [37] presented a rigorous runtime analysis for several variants of ACO algorithms. Their work addresses the question of how long it takes until the algorithm finds an optimal solution for a specific problem.

Furthermore, a review on the current status of Multiple Objective Ant Colony Optimization was addressed in [1]. An extended taxonomy of ACO approaches to multiple objective optimization problems was proposed and many existing approaches were reviewed and described using this taxonomy. This taxonomy offers guidelines for the development and use of Multiple Objective Ant Colony Optimization algorithms.

2.2 Data Mining State-of-the-Art Elements

DM is the process of analyzing data in order to discover useful, possibly unexpected, patterns in data[16]. Two of the most important techniques of DM are classification and clustering. A classification model carries out the task of assigning a class label to an unknown input object after it has been trained with several examples from a given training data set. Clustering on the other hand is the partitioning of a set of input data into subsets (named clusters) so that data in the same subset have something in common. In this subsection, we briefly refer to some of the numerous research efforts in the area of DM. A textbook, like [16], is a more detailed recommended informative source on DM.

DM's contribution to scientific community is indisputable. As DM is becoming more popular, it is gaining wide acceptance in a large number of fields such as healthcare, biomedicine, stock market, fraud detection, telecommunication, text and web mining and others [20,52]. In biomedical research, DM research in DNA analysis has led to the discovery of genetic causes for many diseases and disabilities as well as approaches for disease diagnosis, prevention and treatment [22,46]. Additionally, DM for business continues to expand, as e-commerce and marketing becomes mainstream parts of the retail industry.

An approach proposed by Kargupta et al. describes the Collective Data Mining (CDM) approach, which provides a better approach to vertically partitioned datasets [21].

The design of DM languages, the development of effective and efficient data mining methods and systems, the construction of interactive and integrated data mining environments, and the applications of data mining to solve large-scale application problems, are important challenges for both data mining researchers and data mining system and application developers.

2.3 ACO and DM State-of-the-Art

An interesting research area for ACO is the combination with DM methods for classification and clustering decision making tasks. Modeling classification and clustering as graph search problems allows the use of ACO for finding optimal solutions to these DM tasks. Until today, ACO has been combined with DM methods for classification and clustering in a limited number of studies. A brief description of a number of papers is presented here. The reader is encouraged to read the rest of this chapter for more extensive descriptions and examples of the combination of ACO and DM algorithms.

In [19] Jin et al. proposed a classification rule mining algorithm which was based-on ACO. A number of improvements were implemented to intensify classification accuracy and simplicity of the rules. With these improvements, the overall performance of the algorithm is improved and classification predictive accuracy is enhanced.

Wang and Feng [57] proposed an improved ACO for rule mining classification which is called ACO-Miner. The purpose of ACO-Miner is to give efficient classification rules with accuracy and a simpler rule list based on Ant-Miner. Another interesting approach was proposed in [53] by Thangavel and Jaganathan. In this work, an enhanced ACO algorithm, called TACO-Miner, that has as its main purpose to provide classification rules with a simpler rule list and higher predictive accuracy was presented.

Parpinelli et al. [42] proposed the ACO algorithm for discovering classification rules with the Ant-Miner algorithm. Otero et al. [39] presented an extension to Ant-Miner, named cAnt-Miner (Ant-Miner coping with continuous attributes), which incorporates an entropy-based discretization method in order to cope with continuous attributes during the rule construction process. The same research group [40] introduced the cAnt-Miner2 for mining classification rules**.** The cAnt-Miner2 is a more flexible representation of continuous attributes' intervals and deposits pheromone on edges instead of vertices of the construction graph. Recently, Michelakos et al. [32] presented a hybrid algorithm for medical data mining, combining the cAnt-Miner2 and the mRMR feature selection algorithms.

In addition to the above research efforts, data clustering techniques have also been combined with ACO methods to discover the optimal solution to a number of problems. The classical clustering methods can be improved when these are combined with the concepts of ACO. More specific, the Ant K-Means algorithm modified the familiar K-means clustering algorithm by the probability of locating the objects in a cluster with the use of pheromone, while the rule of this update is obeying the Total Within Cluster Variance [25].

Tsai et al. [56] proposed a new ACO algorithm with a different favorable strategy, namely ACODF (Ant Colony Optimization with differently favorable strategy) by utilizing the main aspects of the classical Ant System. This algorithm exploits the well-known tournament selection strategy to choose the desired path for the clustering problem.

Chelokar et al. presented an ACO technique for clustering, using a matrix of pheromone values as a kind of adaptive memory, which directs other ants towards the optimal clustering solution [6]. Recently, Tiwari et al. [55] proposed two new techniques which slightly improve the general ACO algorithm for Data Clustering. The first technique avoids stagnation by initializing the pheromone values every 50 iterations, and the second technique, again initializes the pheromone values when there is no change on the path after 10 iterations.

3 Ant Colony Optimization

ACO was inspired by the observation of the behavior of real ants. Ant colonies consist of individuals ants with simple behavior, not capable to solve complex problems. However, at the collective level, these societies are capable of solving complex tasks, such as constructing optimal nest structure, or finding the shortest path to food source. Building of chains of ants [26], or formation of drops of ants [54] have been observed.

As it was briefly outlined in Section 2, when ants walking to a food source (Figure 1, state 1) from their nest following a way x, they deposit on the ground a chemical substance called pheromone. The pheromone deposited on the ground forms a pheromone trail y (Figure 1, state 1) which allows the ants to find food sources that have been previously identified by other ants and by following the path with the greatest amount of pheromone laid upon it. Pheromone trails evaporate if more ants do not come along to reinforce their strength. The ants that find the shortest route to the food will arrive back at the nest quicker than the others and will have laid more pheromone along this shortest path (Figure 1, state 2). Therefore, when new ants seek to travel to the food source, since they are guided by the amount of pheromone on the path, they will take the shortest route. It has been observed that eventually all foraging ants converge on the shortest path to a food source (Figure 1, state 3).

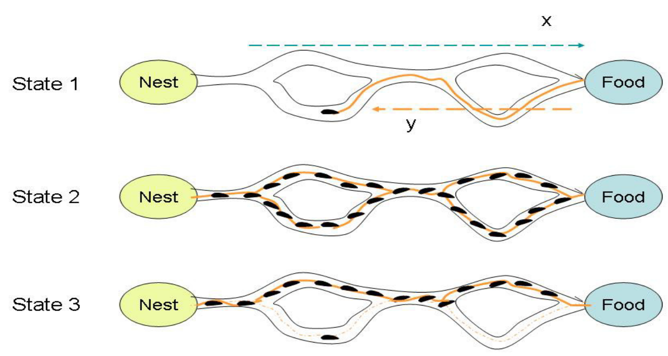

Fig. 1 Food finding procedure followed by ants.

The French entomologist Pierre-Paul Grasse used the term stigmergy [15] to describe this particular type of indirect communication in which "the workers are stimulated by the performance they have achieved." The term is derived from the Greek words stigma (mark, sign) and ergon (work, action), and captures the notion that an agent’s actions leave signs in the environment, signs that other agents sense and that determine and incite their subsequent actions. Researchers investigated experimentally this pheromone laying and following behavior to better understand it.

The first ACO algorithm was published by Marco Dorigo under the name of Ant System (AS) [12]. The algorithm was initially applied on the Travelling Salesman Problem (TSP), where a salesperson wants to find the shortest possible trip through a set of cities on his/her tour of duty, visiting each and every city once and only once. The problem can be viewed as a weighted graph containing a set of nodes N representing the cities the salesperson has to visit. The cities are connected by set of edges E and the goal is to find a minimal-length closed tour of the graph.

In AS, *m* ants ($m \leq n$, where n is the number of the cities) build solutions to the TSP by moving on the problem graph from one city to another until they complete a tour.

For each ant, the transition from city *i* to city *j* at iteration *t* of the algorithm depends on:

1. Whether or not the city has been visited or not. A memory is maintained for each ant to hold the set of cities already visited in the tour which, in turn can be utilized to gather information about the cities that are to be visited when it is in the city *i*.
2. The inverse of the distance from city *i* to city *j*, $n_{ij} = 1/d_{ij}$, (d_{ij} expresses the distance from city *i* to city *j*, or in other words the weight of the edge from city *i* to city *j*) that is called visibility. Visibility is based on the local information and represents the heuristic desirability of choosing city *j* when in city *i*.
3. The amount of pheromone trail $\tau_{ij}(t)$ on the edge connecting city *i* to city *j*. The pheromone is updated as the ant moves from one city to another and represents the learned desirability of choosing city *j* when in city *i*. This update is performed as follows:

$$\tau_{ij} \leftarrow (1-\rho)\cdot\tau_{ij} + \sum_{k=1}^{m}\Delta\tau_{ij}^{k} \tag{1}$$

where ρ in (0,1] is the evaporation rate of pheromone, and $\Delta\tau_{ij}^{k} = \frac{1}{L_k}$ is the quantity of pheromone laid on edge (*i,j*) by the *k*-th ant in the case of the *k*-th ant used the edge (*i,j*) in its tour, otherwise this quantity equals to 0 (where L_k is its tour length).

The probability that the *k*-th ant will choose the city *j* as its next travel point is defined by a probability function. This function applied for ant *k* currently at city *i* during iteration *t* is of the form:

$$P_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{a}\cdot[n_{ij}]^{\beta}}{\sum_{k\in A_k}[\tau_{ik}(t)]^{a}\cdot[n_{ik}]^{\beta}} \tag{2}$$

In this expression the set A_k is the currently valid neighborhood for this ant, i.e. the set of cities not yet visited. This probability function is a combination of two components: the first is the strength of the pheromone trail and the second is a distance decay factor. If α=0 then the pheromone component has no impact and the probabilistic assignment is based on whichever city is closest, whilst if β=0 assignment is simply based on pheromone trail strength, which has been found to lead to stagnation of the solutions giving sub-optimal tours.

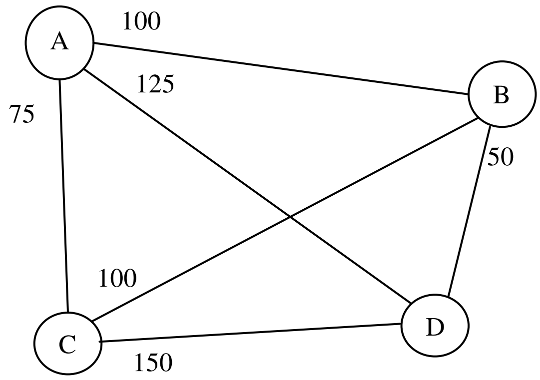

Fig. 2 A four city TSP problem

For example, consider the weighted graph for 4 cities as shown in Figure 2. The distance between cities is denoted along the edge connecting two cities. The first ant starts from city A and has to choose probabilistically one of the three remaining cities to visit, as shown in Figure 2. It does so according to the transition rule given in Equation (2). In the equation, α is set to 1, β is set to 2 and ρ is chosen to be equal to 0.1 [13]. The initial pheromone is set to be equal to 1. The probability the ant will choose the city B is:

$$P^1_{AB}(1) = \frac{(1/100)^2}{(1/100)^2 + (1/125)^2 + (1/75)^2} = 0.293$$

Similarly the probabilities for choosing cities C and D are:

$$P^1_{AD}(1) = 0.187$$

$$P^1_{AC}(1) = 0.520$$

Subsequently the ant chooses to visit city C as its next station. Continuing the iteration, the ant completes the tour by visiting the city B and then the city D. After completing the tour, the ant lays pheromone along the path of the tour. The amount of pheromone added is equal to the inverse of the total length of the tour.

$$\Delta\tau^1_{AC}(1) = \frac{1}{75+100+50+125} = 0.0029$$

The new pheromone levels are calculated using the Equation (1).

$$\tau_{AC}^{1}(1) = (1-0.1)\cdot 1 + 0.1\cdot(0.0029) = 0.90029$$

and,

$$\tau_{AB}^{1}(1) = (1-0.1)\cdot 1 + 0.1\cdot 0 = 0.9$$

Pheromone is updated in the same way to the other edges in the path. Finally the pheromone is decreased along all the edges in order to simulate the pheromone decay. You can see all the new values of pheromone level in the Figure 3. The next ants will start from the remaining cities (second ant will start from city B etc.) and will follow the same procedure in order to complete their tour. The pheromone updates will be done as earlier. The algorithm continues to find the shortest path until a certain number of solution constructions, fixed at the beginning of the algorithm is met. This number is also the terminating condition of the algorithm.

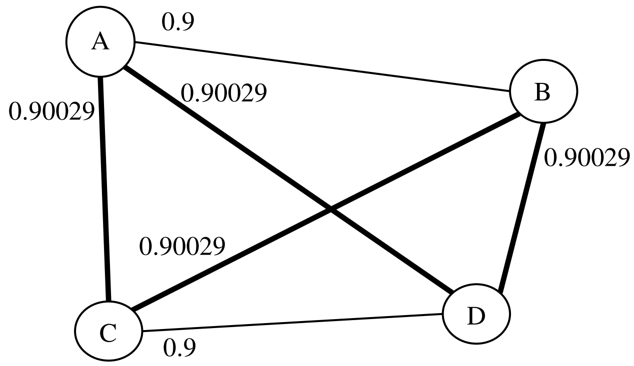

Fig. 3 Pheromone values for the graph in Figure 2 after the first ant finishes a tour.

Summing up, we could say that in order to design a new ant algorithm for a complex combinatorial problem, the problem can be modeled as a search of artificial ants for a best path through a graph. This graph consists of nodes and edges, where nodes represent the basic elements of a solution to the problem and each node is associated with an edge which measures the quality of a partial solution.

An ACO algorithm should have the following basic characteristics:

- an appropriate *problem representation* is required that allows the artificial ants to incrementally build a solution using a probabilistic transition rule. In AS for example the artificial ants build their solution for the TSP by moving on the problem graph from one city to another until they complete a closed tour;
- a *local heuristic* provides guidance to an ant in choosing the next node for the path it is building. This heuristic is problem dependant and for AS it is the inverse of the distance between two cities;

- a probabilistic *transition rule* which determines which node an artificial ant should visit next. The transition rule is dependent on the heuristic value and the pheromone level associated with an edge joining two nodes;
- a *constraint satisfaction* method that forces the construction of feasible rules and in the case of AS [12], an ant must visit each city once and only once during its solution construction;
- a *fitness function* which determines the quality of the solution built by an artificial ant. For the AS algorithm the ant that produces a closed tour of minimal length has the greatest quality, and finally;
- a *pheromone update rule* which specifies how the modification of the pheromone trail laid along the edges of the graph will happen. The pheromone levels are an essential part of the transition rule mentioned above.

Since the first publishing of the Ant System algorithm by Dorigo several versions of the ACO strategy have been proposed, but they all follow the same basic ideas:

- search performed by a population of ants
- incremental construction of solutions
- probabilistic choice of solution components based on stigmergic information
- no direct communication between the ants

Some of the most popular variations of the ACO algorithms other than the Ant System [12] are the Elitist Ant System [58], the Max-Min Ant System (MMAS) [51], the Rank-based Ant System (ASrank) [44] and the Continuous Orthogonal Ant Colony (COAC) system [17].

An ant colony system simulates the behavior of real-world ant colonies since artificial ants have preference for trails with larger amounts of pheromone, shorter paths have a stronger increment in pheromone and ants communicate indirectly with other ants in order to find the shortest path. On the other hand, Parpinelli et al. (2001) [41], showed that there are also some differences between real ants and artificial ants, such as that artificial ants have memory, they are completely blind and time is discrete.

4 Data Mining

The most significant reason which guided DM as a key research and practical area in Information Technology is the wide availability of a vast amount of data. Such data, combined with the availability of a variety of database clusters and other storage facilities, could be utilized to extract valuable pieces of information [16], which in turn could be used in a majority of industrial and scientific areas (e.g. Health, Finance, Marketing etc.).

DM has attracted, throughout the last two decades, a lot of attention and a great number of tools, techniques and algorithms, have been applied in unprocessed data, in order to discover new association rules, predict the outcome of an event, or describe, in convenient ways – e.g. patterns, unsolved problems.

DM is nowadays widely acknowledged as part of the overall Knowledge Discovery process (KDD) [31]. More specifically as stated in [31] the whole KDD process consists of three main phases, the phase of data pre-processing, the phase of data processing (DM) and the phase of data post-processing. DM process, depending on the task performed, may use two data types, namely labeled and unlabeled data. The first type of data contains a class attribute for each data item and mainly appears in training data sets used for classification, whereas in the second type of data no information exists about the attribute class and mainly appears in data sets to be clustered. DM that uses labeled data is characterized as supervised learning, contrary to DM performed upon unlabeled data which is characterized as unsupervised learning. In the remaining part of this section, a brief description of the two main techniques in the DM process, classification and clustering, is given.

4.1 Classification

Classification is a common task of the DM emerging field. With classification, data is arranged into predefined groups with certain characteristics [16]. For example you may use classification to predict whether a given patient is normal, or suffers from breast cancer.

Classification uses labeled data for creating its model. Each data object of the training data set has been allocated to exactly one class, which is described by a specific attribute, the class label attribute. The classification data model that is derived by considering this allocation can be in turn used to classify new data items (without the class label attribute) and more generally, to extract useful pieces of information, valid patterns, predict future trends, etc..

There exist numerous techniques for data classification. For more information the reader is encouraged to study [16,31]. The main most used techniques are briefly outlined here:

- *Decision trees:* A classical tree-structure flowchart, where starting from the root node of the tree, progression is made to the internal nodes, which represent a test on the value of one, or more data attributes. A decision is obtained, when a node representing no test is reached.
- *Association Rules:* A set of rules having type «if Condition then Prediction» where the Condition could be a conjunction of terms and the derived Prediction could be a possible solution that satisfies the Condition.
- *K-Nearest neighbors algorithms:* The training samples are portrayed by dimensional numeric attributes and with the use of the Euclidean distance between two samples, the K samples which are closest to the unknown sample are identified, and the most common class among them is identified.
- *Artificial Neural Networks:* A composite modeling technique based upon the model of a human neuron. The system made consists of simple parallel-functioning interconnected units (artificial neurons) that form a network called a neural network). The operations carried out by these units conclude to the prediction of one, or more events.

4.2 Clustering

Clustering, on the contrary, is an unsupervised learning technique, as it is performed upon unlabelled data and primarily depicts a method where objects of similar characteristics are grouped together to form clusters. Clustering mainly aims in forming the amount of unmanaged data to manageable piles, by discovering homogeneous groups. Clustering has numerous applications. For example, by using past buying records, clustering can be used for determining groups of customers with similar behavior, for marketing purposes. A basic discrimination of clustering techniques is presented below:

- *Hierarchical Clustering*: Basic type of the clustering methods, where a tree of classes is build, called a dendrogram. The fundamental idea for the tree is to start with each object in a cluster of its own and merge the closest pair of clusters, ending up in one cluster, enclosing everything.
- *Non-hierarchical Clustering*: In this type of clustering technique, classes which are not subclasses of each other are built. The fundamental technique representing non-hierarchical clustering is the k-means algorithm. The k-means algorithm uses the concept of a centroid, the median point in a group of points. Briefly, values of k points as the initial centroids are chosen, then an assignment for every object to the nearest to the centroid cluster is made, a recalculation for the centroids of the k clusters is performed and finally the last two steps are repeated until the centroids remain unaffected.

Several algorithms have been proposed in order to perform clustering techniques upon data. The selection of the appropriate algorithm to be used depends mainly on the type of data which is offered, as well as, on the particular purpose or the application that DM is applied to [16]. In [16] several clustering techniques are exhaustively explained and paradigms of the techniques are outlined, therefore the reader is encouraged to study further the Cluster Analysis chapter of [16], in order to acquire supplementary details.

5 Ant Colony Optimization and Data Mining Techniques

5.1 Data Classification and Ant Colony Optimization

The basic elements of the solution to the classification rule induction problem are the attribute terms. ACO algorithms used for classification aim to discover knowledge expressed in the form of IF-THEN classification rules: IF (conditions) THEN (class), where conditions follow the form ($term_1$) AND ($term_2$) AND ... AND ($term_n$). The class to be predicted by the classification rule is represented by the THEN part corresponding to the rule's consequent and the IF part corresponds to the rule's antecedent. An instance that satisfies the IF part will be assigned the class predicted by the rule. Each term in the rule is a triple (attribute, operator, value), such as <smoke=no>. The value is a value that belongs to the domain of the attribute. For example a simple rule for a weather dataset (Table 1) containing four predicting attributes namely outlook with three possible values {sunny, overcast, rainy}, temperature with three possible values {hot, mild, cool}, humidity with two possible values {high, normal} and windy with two possible values {true, false} concerning the problem of playing outside, or not (attribute play with two possible values {play, don't play}) could be IF <humidity = normal> THEN <play>.

An attribute term, $term_{ij}$, is in the form $A_i = V_{ij}$, where A_i is the i-th attribute and V_{ij} is the j-th value of the domain of A (e.g. humidity is the third attribute and normal is its' second possible value in the above example) . Terms of a predicting attribute and class attribute are called predicting terms and class terms, respectively (e.g. <humidity = normal> is a predicting term and <play=yes> is a class term in the above example). The process of construction of a rule is to search for a combination of predicting terms in the rule antecedent that best identifies a class term. Therefore, in the graph of a classification rule induction problem, the nodes represent attribute terms (e.g. <humidity=normal>) and edges model the quality of the attribute terms. An artificial ant then constructs a rule by visiting a set of possible nodes in the graph and forms a path that ends at a class term node (e.g. <play=yes>). A complete path is a constructed rule. The quality of the path is assessed by a global fitness function. The quality of a node is evaluated by a heuristic value and a pheromone level value associated with the node. These values provide a guide to the ant for which node should be visited next.

Table 1 Weather dataset subset of 10 instances [47].

Outlook	Temperature	Humidity	Windy	Play
Sunny	Hot	High	false	Don't Play
Sunny	Hot	High	true	Don't Play
Overcast	Hot	High	false	Play
Rain	Mild	High	false	Play
Rain	Cool	Normal	false	Play
Rain	Cool	Normal	true	Don't Play
Overcast	Cool	Normal	true	Play
Sunny	Mild	High	false	Don't Play
Sunny	Cool	Normal	false	Play
Rain	Mild	Normal	false	Play
Sunny	Mild	Normal	true	Play
Overcast	Mild	High	true	Play
Overcast	Hot	Normal	false	Play
Rain	Mild	High	true	Don't Play

Parpinelli et al. (2002) [42] proposed the ACO algorithm for discovering classification rules with the Ant-Miner algorithm. Starting from a training dataset, Ant-Miner generates a set of ordered rules through iteratively finding a "best" rule that covers a subset of the training data, adds the "best" rule to the induced rule list, and then removes the examples covered by the rule (e.g. the rule <humidity=normal> then <play=yes> covers six examples of the dataset given in Table 1), until a stop criterion is reached.

In Ant-Miner, an artificial ant follows three procedures to induce a rule from a current training dataset. Rule construction, rule pruning and pheromone updating. The artificial ant starts from an empty rule (no attribute terms in rule antecedent), and selects one attribute term, at a time, adding to its current partial rule based on the local problem-dependent heuristic value and the pheromone level associated with the term. Terms with higher heuristic value and pheromone level are preferred, and terms whose attributes are already present in the current rule antecedent are not considered. Two constraint rules must be satisfied when the ant selects a term. The first one is that two terms that belong to the same attribute must not appear in a rule and the second one is that a rule must cover at least a predefined minimum number of examples. In order to satisfy the first restriction, artificial ants must "remember" which terms are contained in the current partial rule. The second restriction helps to avoid over-fitting and improves the generality of a rule and should be satisfied both in rule construction and in the rule pruning process.

The construction stops when adding a term would make the rule coverage (the number of examples the rule covers) smaller than a user-specified threshold, or until all attributes have been used. The local heuristic function applied in Ant-Miner is an entropy measure of individual terms and is defined by:

$$H(C \mid A_i = V_{ij}) = -\sum_{c=1}^{k} (P(c \mid A_i = V_{ij}) \cdot \log_2 P(c \mid A_i = V_{ij})) \tag{3}$$

where:

- C is the class attribute and k is the number of class values,
- A_i is the i-th attribute and V_{jj} is the j-th attribute value of the i-th attribute,
- $P(c \mid A_i = V_{jj})$ is the probability of observing class c conditional on observing $A_i = V_{jj}$.

For example, the entropy of the term "outlook = rain" in the training data in Table 1 using the Equation (3) is:

$$H(Play \mid outlook = rain) = -\frac{3}{5} \cdot \log_2(\frac{3}{5}) - \frac{2}{5} \cdot \log_2(\frac{2}{5}) = 0.97$$

The higher the entropy value of a term, the more uniformly distributed the classes are and, so, the smaller the probability that the current ant chooses this term to add to its partial rule. However, the ant prefers to choose a term with higher heuristic value. It, therefore, requires a proper normalization of the entropy values, which is handled by a normalized heuristic function:

$$n_{ij} = \frac{\log_2 k - H(C \mid A_i = V_{ij})}{\sum_{i=1}^{a} x_i \cdot \sum_{j=1}^{b_i} (\log_2 k - H(C \mid A_i = V_{ij}))} \tag{4}$$

where:

- α is the total number of attributes,
- x_i is set to 1 if the attribute A_i is not yet selected; otherwise, it is set to 0,
- b_i is the number of domain values of the i-th attribute.

For example, the heuristic value for the term "outlook = rain" in the training data in Table 1 using the Equation (4) is:

$$\eta_{(outlook=rain)} = \frac{1-0.97}{(1-0.97)+(1-0)+(1-0.97)+(1-0.906)+(1-0.811)+(1-0.984)-(1-0.65)+(1-0.811)} = 0.0158$$

The Ant-Miner [41] uses the transition rule given in Equation (5). Given an attribute-value pair, the transition rule gives the probability of adding the attribute value pair to the rule. The probability is calculated for all of the attribute- value pairs, and the one with the highest probability is added to the rule.

$$P_{ij} = \frac{n_{ij} \cdot \tau_{ij}(t)}{\sum_{i=1}^{a} x_i \sum_{j=1}^{b_i} (n_{ij} \cdot \tau_{ij}(t))} \tag{5}$$

where:

P_{ij} is the probability that $term_{ij}$ is selected for addition to the current partial rule antecedent with a range [0,1], η_{ij} is the heuristic value associated with $term_{ij}$,

- $\tau_{ij}(t)$ is the amount of pheromone associated with a $term_{ij}$ at iteration t,
- α is the total number of attributes,
- b_i is the number of domain values of the i-th attribute,
- x_i is set to 1 if the attribute A_i is not yet selected; otherwise, it is set to 0,

Once the artificial ant stops building a rule, the majority class among the examples covered by the rule antecedent is then assigned to the rule consequent.

After constructing the rule, the artificial ant performs the rule pruning procedure. The purpose of rule pruning is to increase the quality and comprehensibility of the rule built by simplifying the rule antecedent. This is done by iteratively removing one term at a time from the rule antecedent while the quality of the rule is improved. The quality of a rule, denoted by Q, is defined by the following formula:

$$Q = \frac{TP}{TP + FN} \cdot \frac{TN}{FP + TN} \tag{6}$$

- *TP* (true positive) is the number of examples covered by the rule that belong to the class predicted by the rule,
- *FP* (false positive) is the number of examples covered by the rule that belong to a class different from the class predicted by the rule,
- *FN* (false negative) is the number of examples that are not covered by the rule, but belong to the class predicted by the rule,
- *TN* (true negative) is the number of examples that are not covered by the rule and that do not belong to the class predicted by the rule.

For example, the quality of a rule, IF <outlook = sunny> AND <humidity = high> THEN <don't play>, of the training data in Table 1 is:

$$Q = \frac{TP}{TP + FN} \cdot \frac{TN}{FP + TN} = \frac{3}{3+0} \cdot \frac{3}{0+3} = 1$$

This fitness function evaluates the accuracy of a rule without considering rule simplicity. The accuracy consists of both accuracy among positive examples (called sensitivity) and accuracy among negative examples (called specificity). The range of Q values is in [0, 1]. In each iteration of rule pruning, every term in turn is temporarily removed from the rule, a new rule consequent is assigned and the quality of the rule is reconsidered. At the end of the iteration, only the term whose removal improves the rule quality most is actually left out. The rule pruning process stops when the removal of any term does not improve the rule quality or the rule has just one term. Once rule pruning is done, the artificial ant increases the pheromone level of a term in the rule antecedent according to the rule quality given by the following formula:

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \tau_{ij}(t) \cdot Q \tag{7}$$

where:

- $\tau_{ij}(t)$ is the pheromone level of the $term_{ij}$ at iteration t,
- Q is the quality of the constructed rule.
- i,j belong to the constructed rule

For example, if the ant adds the rule IF <outlook = sunny> AND <humidity = high> THEN <don't play>, from the training data in Table 1, then the pheromone value at these nodes is:

$$\tau_{ij}(2) = \tau_{outlook=sunny}(1) + \tau_{outlook=sunny}(1) \cdot 1 = 2$$

In our example the initial pheromone level, in favor of simplicity, is equal to 1 but the actual initial pheromone level for each term is given by the type:

$$\tau_{ij}(t=0)=\frac{1}{\sum_{i=1}^{a} b_i}$$

where:

- α is the total number of attributes and
- b_i is the number of domain values of the i-th attribute.

The ant then normalizes the pheromone levels of all terms (each pheromone level is divided by the sum of all pheromone levels) which reinforces the pheromone levels of the terms occurring in the rule antecedent and decreases the pheromone levels of other terms that are not selected in the rule.

These procedures (rule construction, rule pruning and pheromone updating) by which an artificial ant induces a rule, are repeated until every artificial ant (number of ants is a user-defined parameter) has generated a rule, or the current rule has been generated by previous (maxRulesConvergence - 1) ants. The maxRulesConvergence is a user-defined parameter for testing the convergence of ants, which simulates the convergence of real ants to the shortest path between a food source and their nest.

The best rule among the rules generated by all ants is added to the induced rule set. The training dataset is appropriately updated by removing all the examples covered by the best rule. Ant-Miner uses this updated training dataset to induce a new rule that will be added to the rule set through the process described above. Different training datasets are different problems, similar to different food sources that real ants tackle, and, so, the pheromone level of terms needs to be re-initiated. In the end, Ant-Miner stops when the number of examples in the training dataset is smaller than a user-defined threshold (MaxUncoveredCases).

A significant difference between Ant-Miner and other ACO algorithms is the size of the population of ants required between two pheromone updates. Ant-Miner works with a population of a single ant. This ant constructs a rule and updates pheromone levels according to the quality of the rule. Other ACO algorithms normally require a group of artificial ants to work together, such that each ant finds a solution and the pheromone is updated according to the best solution among the solutions found.

Ant-Miner employs an ACO approach providing a mechanism for conducting a global search which is more effective than those provided by traditional covering algorithms. Analogous to the application of a genetic algorithm to classification rule induction, Ant-Miner copes better with attribute interaction than greedy rule induction algorithms do.

Ant-Miner, however, has some limitations. One of the limitations is that Ant-Miner supports only nominal (categorical, or discrete) attributes where the only valid relational operator is "=" and in a preprocessing step continuous attributes need to be discretized using other techniques, such as the C4.5- Disc discretization method [28].

Following the main aspects of Ant-Miner, a number of ACO variations were proposed. They involve different pruning and pheromone update procedures, new rule quality measures and heuristic functions, for discovering fuzzy classification rules, rules for multi-label classification problems and handling of continuous attributes. A typical example of an Ant-Miner variation able to cope with continuous attributes is the cAnt-Miner2 algorithm [40].

5.2 Data Clustering and Ant Colony Optimization

The basic model for data clustering techniques based on ideas coming from Ant Colonies was firstly introduced by Deneubourg et al. (1990) [10]. The main idea behind their method comprises the basic activities of an ant colony to gather items in order to form piles e.g. cluster dead bodies and sort them discriminating among different kind of items. The model proposed is a continuous model, where ants are represented as simple agents, which randomly move into a two-dimensional (square) grid, with a number of limitations in order to pile their corpses. Items distributed within such an environment could be picked-up with a probability $P_p = (\frac{a_1}{a_1 + f})^2$ or dropped-down with a probability $P_d = (\frac{f}{a_2 + f})^2$.

In each iteration step an ant explores its neighborhood and computes the above probabilities. Parameters a_1 and a_2 are threshold constants and their values are compared to the value of function f that denotes a high probability of picking up or dropping down an item. For example, if a_1 is much higher than f, then P_p converges to 1, thus making the probability of an ant to pick-up an item quite high. Function f is a function that encapsulates the notion of the average distance of elements [10].

This procedure is influenced by a number of parameters, within the agents' local neighborhood which are set empirically and may produce more clusters than the optimal number. Moreover, in the basic model the absence of pheromone could be critical in a number of cases. For that reason many improvements to this algorithm have been proposed. The main extension of Deneubourg's model was introduced by Lumer and Faieta (1994) [30] who use a continuous similarity function and define the idea of distance, or dissimilarity d between objects in the space of object attributes. This technique has been called *Ant Colony Clustering* and a variety of modifications have been proposed, which modify existing parameters, or introduce the notion of pheromone in the algorithm in order to reduce the large amount of computational time, or improve convergence of the algorithm [3,34].

A novel approach presented by Tsai et al. (2004) [56] is not only based on ideas coming from Ant Colonies, but utilizes the classical Ant System and proposes a new ACO algorithm with a different favorable strategy (Ant Colony Optimization with differently favorable strategy - ACODF). This algorithm initially uses favorable ants in order to provide a solid solution for the clustering problem, and then it uses simulated annealing in order to decrease possible paths and finally exploits the well-known tournament selection strategy to choose the desired path.

The basic steps of the algorithm are summarized in the following. In the initialization phase n data points are chosen and m ants are assigned to m nodes (n represents the number of nodes and m the number of ants). Then, a computation is performed concerning the number of nodes that ants should visit (initially for the first time and later randomly for each ant in arbitrary directions). Afterwards, a random selection of a number of trails is performed and with the aid of a selection mechanism (Tournament Selection in this case) the algorithm finds the pheromone trail with high quantity. In the next step this pheromone quantity of every trail is updated and an iteration of the above steps is executed, until all trails of pheromone quantity reach a stable state. In the last step, clustering is performed using the value of pheromone quantity.

Moreover, the results obtained with ACODF algorithm [56] were compared with two other well-known approaches for data clustering, Fast Self-Organizing Map (FSOM) combining K-means (aka FSOM+K-means) and Genetic K-means algorithm (GKA). The comparison showed that ACODF algorithm performs better in terms of time cost when the data sets used are data sets of 300 and 579 samples and the clustering methods used are both non-spherical and spherical. Additionally, ACODF produces a smaller number of errors (better clustering results) than the two other algorithms.

Other approaches include the improvement of classical clustering methods when these are combined with the concepts of ACO. The major paradigm of such an approach is presented in [25] where the *Ant K-Means* algorithm is introduced, which modifies the familiar K-means clustering algorithm by the probability of locating the objects in a cluster with the use of pheromone, while the rule of this update is according to the Total Within Cluster Variance (TWCV). The main disadvantage of techniques based on Ant K-means algorithm and its variations is that the number of the clusters and the corresponding centroids should be known in advance and are generated with the aim of the Ant System-based Clustering Algorithm (ASCA) which was also developed by the authors.

This algorithm consists of four sub-procedures (*divide, agglomerate_obj, agglomerate* and *remove*) and calculates the TWCV. The main algorithm introduced modifies the well-known K-means algorithm in the way the location of objects in a cluster is calculated and the probability used is modified by the pheromone (updating pheromone according to TWCV). The first step of AK (Ant K-means) algorithm is the initialization phase, where all the parameters including the number of clusters and its centroid are initialized. In the second step, equal amount of pheromone is laid on each path, and then each ant chooses the centroid with probability *P,*

$$P_{ij}^{k} = \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{c}^{nc}\tau_{ic}^{a}\eta_{ic}^{\beta}},$$

where i is the start point, j is the end point which the ant k chooses eventually to move-in, c is the centroid and nc is the overall number of the centroids. The next step, is the update of pheromone by

$$\tau_{ij} \leftarrow \tau_{ij} + \frac{Q}{TWCV},$$

where Q is a constant as described in [25]. Afterwards, a calculation of the object $O_{\text{center}}(T_k)$ which is the center of all objects in T, where k=1,2,3,..*nc* is performed and a recalculation of TWCV is performed, if necessary. Parameter T describes the set which includes all used objects (maximal number is n). If TWCV is changed, probability P is recalculated in the third step. The final step is to run the procedure *Perturbation* in order to leap from the local minimal solution and if the number of iterations is accomplished the algorithm is stopped, otherwise P is re-calculated.

The solution proposed in [25] is analytically compared with two other methods (Self Organizing Maps + K-means and Genetic K-means algorithms) via data sets which are generated by the Monte Carlo simulation. Moreover, another comparison is performed upon real case data (suggestions formulated for the price reduction in plasma TV's).

The list of clustering approaches using ACO incorporates one more approach, which is described in [6]. In this approach a matrix of pheromone values is used as a kind of adaptive memory, which directs other ants towards the optimal clustering solution. The algorithm outlines a methodology used to discover an optimal clustering technique in order to assign N objects to one of the K clusters. The pheromone matrix used is of size N x K, thus each object is associated with K pheromone concentrations. The matrix is updated during each iteration depending on the solutions produced. Initially, each ant starts with an empty solution string S and in order for a solution to be constructed, the agent utilizes the pheromone trail information to assign each element that resides in S to an appropriate cluster label.

During the first iteration of the algorithm each element of the matrix is initialized to the same values. As the algorithm carries on, the pheromone matrix is updated accordingly, depending upon the solutions produced. At each iteration, the agents or software ants produce trial solutions using pheromone trails in order to obtain the optimal or near-optimal partitioning of the given N objects into K clusters (groups). Subsequent to the generation of the trial solutions, a further improvement of the solutions proposed is achieved, by performing a local search. The pheromone probability is used in order to choose among the various clusters and is given by

$$p_{ij} = \frac{\tau_{ij}}{\sum_{K=1}^{\mathrm{K}} \tau_{ik}},$$

where p_{ij} is the normalized pheromone probability for element i that belongs to cluster j, and j=1,2…K.

Recently, Tiwari et al. (2010) [55] proposed two new techniques which slightly improve the general ACO algorithm for Data Clustering. In the generalized model, each agent initially begins with an empty solution string S and a pheromone matrix τ which maintains the ant's position in a specific cluster and is initialized to a small value τ_0. As the algorithm proceeds, the agent uses the pheromone trail information obtained during each iteration, in order to update the pheromone matrix τ and extend the solutions produced which show the probability of an ant that belongs to a specific cluster. Later on, a local search is performed, which re-organizes the pheromone matrix depending on the quality of the solutions produced. In order to optimize the solutions produced, the objective function, which is defined as the sum of squared Euclidian distances between each object and the center of belonging cluster, should be minimized. After a number of iterations is performed, the solution which has the lowest function value is chosen as the optimal solution. The first proposed technique in order to avoid stagnation, initializes the pheromone values every 50 iterations, and the second technique, again initializes the pheromone values when there is no change on the path after 10 iterations. This solution describes the optimal partitioning of objects of a given dataset into several groups [55].

6 Applications and Examples

Bursa and Lhotska (2007) in their work [4] describe the way clustering techniques use ant colonies. They used the ACO_DTree method [5] (a method based on the MAX-MIN Ant System algorithm [51]) together with Particle Swarm Optimization as a local search method. Their study examined two types of biological signals. Electrocardiograms (ECG) and Electroencephalogram (EEG). Electrocardiograms (ECG) an electrical recording of heart activity is one of the most important diagnostics techniques used in patients. Its processing consists of seven stages: signal pre-processing, signal transfer and/or storage, digital signal processing and feature extraction, clustering of the similar data, signal classification and expert validation. From the ECG signal, eight features have been automatically extracted [8] and two classes have been used (normal cardiac action and abnormal cardiac action) for the above mentioned study. Electroencephalogram (EEG) is an electrical recording of brain activity which is used in order to classify stages of sleep. The EEG recordings used contain eight EEG channels, Electrooculogram (EOG), Electromyogram (EMG), Respiratory channel (PNG) and Electrocardiogram (ECG). All these recordings have been classified by a medical expert into four classes (wake, quiet sleep, active sleep, movement artifact).

In the first stage of the ACO_DTree method, a population of random solutions is generated. In each iteration of the algorithm, the population is enhanced with new solutions driven by pheromone levels. The new updated population is evaluated and only a specified number of solutions is preserved. Pheromone updating is made by depositing a certain amount of pheromone balanced to the quality of best individuals and afterwards by pheromone evaporation. The matrix used for the pheromone updating procedure conforms to a full graph where nodes represent feature indexes and edges contain pheromone representing transition from one feature to another. Only the best solutions deposit an amount of pheromone, determined by the quality of the solution, into this matrix. This process is iterated up to maximum level of the tree and finally the trees are optimized using a local search technique (the Particle Swarm Optimization method which is a population approach inspired by the behavior of animals with swarm intelligence).

Data from the MIT-BIH database [14] with more than 80.000 for ECG and about 450.000 for EEG records were used for this study. The hybrid combination of DM algorithms for data partitioning and data classification with ACO allows better convergence leading to increased robustness and clearer structure with better clinical use [4].

Another interesting application presented by Kuo et al. (2007) [24] concerns a framework which integrates both the clustering analysis and association rules mining to discover the useful rules in the database through an ACO system. The first component of the proposed method is the clustering analysis and the second one is the association rules mining. The first stage employs the ant system-based clustering algorithm (ASCA) and ant K-means (AK) [25] to cluster the database, while the ant colony system-based association rules mining algorithm is applied to discover the useful rules for each group. The main reason for clustering the database is that this can dramatically decrease the mining time. In order to assess the proposed method, a database being provided by the National Health Insurance Plan of Taiwan Government is applied.

After encoding, clustering analysis was done with the two-stage clustering algorithm, which includes ASCA and AK. The application of the algorithm generated three clusters. These clusters were chosen for DM with the ACS-based association rule mining algorithm.

The main target of the application was to develop a decision support system about patient treatments that is able to extract important relationships or association rules between diseases in order to provide an alternative way to help diagnose the diseases and to specify treatments for them. Such a system could help the physicians pay more attention on important groups of patients and find out the hidden relation in these groups easier.

The computational results showed that the proposed method not only can extract the useful rules faster, but also can provide more precise rules for the medical doctors and let the researchers pay more attention on some important patient groups and find out the hidden relation in the groups easier.

Kumar and Rao (2009) [23] proposed a use of DM algorithms for the extraction of knowledge from a large set of flow shop schedules. In the first section of their work they describe the ACO algorithm used and the method to generate a population of the optimal sequences. The second section of their work deals with mining the solutions given by the ACO algorithm in order to extract from them decision rules. These rules are based on several attributes like processing time, position in the job, remaining time of the job or machine loading. Finally they used a Decision Tree, (See5 classifier –a commercial version of C4.5 [47]) in order to find their affection order of operation on all machines.

Finally, another interesting application was proposed by Phokharatkul et al. (2005) [45]. They presented a system of handwritten Thai character recognition, which is based on the Ant-miner algorithm. The system uses zoning for each Thai character to determine each character and three attributes of each character in each zone are extracted. These attributes are Head zone, End point, and Feature code and are used by the Ant-miner algorithm in order to classify 112 Thai characters (76 alphabet characters and 36 special symbols).

Thai characters are composed of lines, curves, circles and zigzags. The head is normally a starting point of writing a Thai language character. It is one of the distinctive features of Thai characters and it is defined as a circle or a closed loop in a character [7]. The end point is the point that has only one point connected to it [7] and finally the feature code is defined by the maximum number of points that the referent lines pass in its zone [7]. The data used in this application were collected from 100 persons where each person made 3 copies of a sheet with handwritten characters providing a total data set of 33600 characters.

On the first step of the model, each handwritten character is converted to bitmap by a scanner into a two-color bitmap file. On the next step, an algorithm [7] is used to convert each bitmap into a character image that is only 1 pixel in width. Afterward, each character is normalized to 128x128 pixels and segmented into 12, 9 and 15 zones with the same width and height for feature Head, Endpoint, and Feature code. In this Feature extraction step the features of each character are extracted and saved to a file. In the next step, the Ant-Miner algorithm is used for training the recognition system and finally, the data of 11200 samples are used in order to classify the characters into the next five groups (lower, middle and low, middle, middle and upper and upper characters) [45]. Finally data of each group are classified by the Ant-miner algorithm and the induced rule list is used as the recognition engine. The experimental results shown that the system can recognize 97% of the training set.

7 Conclusions

The audience of this chapter includes researchers, instructors, senior students and graduates of higher education, who are interested in next generation data technologies that handle (possibly distributed) data in a collaborative manner. More specifically, in this chapter, we reviewed a technique which is based on simple agents that collaborate in order to solve a problem. This technique was inspired from the physical behavior of real ants and the way they behave in order to solve problems, like finding food or sorting broods. This technique, named ACO, and its collaborative use with two DM techniques, classification and clustering, which are the most widely used tasks in DM, have been outlined. The chapter has focused in making a review of work on the use of ACO for classification and clustering purposes. The enabling technology which is derived from the collaborative use of ACO and DM leads to improved algorithms and techniques with numerous usages, as presented in Section 6 by providing contemporary real-world examples of various application areas e.g. Health, Marketing, Finance, Molecular Biology.

8 Future Trends

The heuristic function, the pheromone updating strategy and the pruning procedure used in an ACO algorithm are among the basic components of an ACO algorithm. These parts of the algorithm influence its performance and their fine tuning, or correct choice could lead to better accuracy. Several papers in the literature propose this tuning as a worthy target, e.g. [19,40]. We believe that, such a tuning, taking into account the respective real application areas, is also important for collaborative ACO-DM algorithms.

Since recently, ACO algorithms were not able to cope with continuous variables and a pre-processing step of discretization was mandatory. Otero et al. in a recent work [40] introduced a new promising algorithm able to cope with such variables, having the necessary discretization procedure embedded on the main algorithm procedure. The encapsulation of a discretization method in the rule construction process of the ACO algorithm used for classification showed that better results can be achieved. As future research direction, it would be interesting to investigate the performance of different discretization methods in the rule construction process.

Besides their main components, ACO algorithms have a number of system parameters that influence their performance and/or accuracy [18,27]. Detailed experimentation is needed to determine the effects of these parameters, and develop an understanding of methods that set parameters appropriately for particular problems. Michelakos et al. [33] recently studied various system parameter settings of the cAnt-Miner2 algorithm [40]. Further experiments, to study the influence of system parameters on the performance of ACO-DM algorithms for particular problem areas have been planned.

Another main issue that has emerged from collaborative ACO-DM algorithms is their computational cost [41]. This cost is extremely high when the search space (number of predicting attributes) is large. Most of the techniques presented in this chapter are dealing with a rather small amount of data (residing in main memory) and mainly with a single dimension. An interesting research direction could be the adaption of such techniques for applying on large amount of data, which (inevitably) reside on disk, in transactional Databases, Data Warehouses, or specialized disk based data structures and / or have more than one dimension. Apart from accuracy of the result, the I/O and CPU performance of such techniques could be studied.

Moreover the application of collaborative ACO-DM techniques on distributed data, resulting from possibly heterogeneous sources, like data streams, requires appropriate data collection and processing methods that aim at high accuracy and/or performance. This is also considered a challenging issue.

New possibilities might result, regarding the improvement of accuracy and/or performance, by the introduction of hybrid ACO techniques and their application for DM tasks. In a recent study [32], a hybrid algorithm for data classification was presented, combining the cAnt-Miner2 [40] and the mRMR feature selection [43] algorithms. The resulting algorithm was very promising and was experimentally compared to the (non hybrid) cAnt-Miner2 algorithm, using public medical data sets.

Another issue that is worth researching is the appropriate (for ACO use) modeling of other DM tasks, or modeling of different approaches to classification and/or clustering [19], since such a modeling is necessary in order to apply ACO to a problem (see Section 3). Since the accuracy of DM techniques is problem and data dependent, the application of ACO-DM techniques to diverse problem areas (related to current, or future applications) and their thorough comparison with other (state-of -the-art) DM techniques would be interesting. In general, a thorough comparison which will encompass a significant number of DM techniques already proposed, including ACO-DM ones, would be very useful and informative.

Finally, increasing attention could be given to even more challenging problems, involving dynamic data (temporal and/or spatio-temporal data) and their constraints. Dynamic problems are characterized by the fact that the search space changes in the course of time. Hence, the conditions of the search, the definition of the problem instance and, thus, the quality of the solutions already found may change while searching. It is crucial in such situations that the algorithm is able to adjust its search direction and follow the changes of the problem being solved, exhibiting (a kind of) self-adaptation.

9 Key Terms

Ant Colony Optimization (ACO): The ant colony optimization algorithm is a probabilistic technique for solving computational problems aiming at finding an optimal path in a graph, based on the behavior of ants seeking a path between their colony and a source of food.

Agent: an autonomous entity which observes and acts upon an environment and directs its activity towards achieving goals

Ant Colony: An ant colony is an underground lair where ants live. Ants are social insects that form colonies which range in size from a few up to millions of individuals.

Attributes: An attribute is frequently and generally a property of a property and can be considered metadata of an object, element, or file. A specification that defines a property.

Categorical (or Nominal) Attributes / Values: A categorical attribute has values that function as labels rather than as numbers. For example, a categorical attribute for gender might use the value 1 for male and 2 for female.

Continuous Attributes / Values: A continuous attribute has real numeric values such as 1, 2, 6.28, or -7. Examples of continuous attributes are blood pressure, height, weight, age.

Classification: Classification is the assignment of a class label to an input object. The term refers to either of the task, the problem of, and the result of such an assignment.

Classification Rule: IF-THEN classification rules are in the form: IF (conditions) THEN (class), where conditions follow the form ($term_1$) AND ($term_2$) AND ... AND ($term_n$).

Clustering: Clustering or cluster analysis is the assignment of a set of observations into subsets (called clusters) so that observations in the same cluster are similar in some sense.

Data Mining: Data mining is the process of analyzing data in order to discover of useful, possibly unexpected patterns in data.

Graph: Graph is a mathematical structure used to model pair wise relations between objects from a certain collection. A "graph" in this context refers to a collection of vertices or 'nodes' and a collection of edges that connect pairs of vertices. A graph may be undirected, meaning that there is no distinction between the two vertices associated with each edge, or its edges may be directed from one vertex to another.

Learning (Supervised): Supervised learning is a machine learning technique for deducing a function from training data. The task of the supervised learner is to predict the value of the function for any valid input object after having seen a number of training examples. One form of supervised learning is classification.

Learning (Unsupervised): In machine learning, unsupervised learning is a class of problems in which one seeks to determine how the data are organized. One form of unsupervised learning is clustering.

Optimization: Optimization refers to choosing the best possible element from some set of available alternatives.

Pheromone: A pheromone is a chemical substance that triggers a social response in members of the same species. Ants use pheromone in order to communicate indirectly.

Swarm Intelligence: Swarm intelligence describes the collective behavior of decentralized, self-organized systems, natural or artificial. These systems are typically made up of a population of simple agents interacting locally with one another and with their environment leading to the emergence of "intelligent" global behavior, unknown to the individual agents.

Stigmergy: Stigmergy is a mechanism of indirect coordination between agents. It is derived from the greek words stigma (mark, sign) and ergon (work, action), and captures the notion that an agent's actions leave signs in the environment, signs that it and other agents sense and that determine and incite their subsequent actions.

Acknowledgments. The first author acknowledges the financial support of "Heraclitus II" program (Operational Program "Life Long Learning"), funded by the European Union and the Greek State.

References

1. Angus, D., Woodward, C.: Multiple objective ant colony optimization. Swarm Intelligence 3(1), 69–85 (2009)
2. Borkar, V.S., Das, D.: A novel ACO algorithm for optimization via reinforcement and initial bias. Swarm Intelligence 3(1), 3–34 (2009)

3. Boryczka, U.: Finding Groups in Data: Cluster Analysis with Ants. Applied Soft Computing 9(1), 61–70 (2009)
4. Bursa, M., Lhotska, L.: Ant Colony Cooperative Strategy in Electrocardiogram and Electroencephalogram Data Clustering. In: Nature Inspired Cooperative Strategies for Optimization (NICSO 2007), pp. 323–333 (2007)
5. Bursa, M., Lhotska, L., Macas, M.: Hybridized swarm metaheuristics for evolutionary random forest generation. In: 7th International Conference on Hybrid Intelligent Systems (HIS 2007), pp. 150–155 (2007)
6. Chelokar, P.S., Jayaraman, V.K., Kulkarni, B.D.: An Ant Colony Approach for Clustering. Analytica Chimica Acta 509(2), 187–195 (2004)
7. Choruengwiwat, P.: Thai handwritten character recognition using extraction of distinctive features. Master's Thesis, Department of Electrical Engineering, Chulalongkorn University, Thailand (1998)
8. Chudacek, V., Lhotska, L.: Unsupervised creation of heart beats classes from long-term ECG monitoring. In: 18th International Conference of European Association for Signal Processing (EURASIP) Biosignals, pp. 199–201 (2006)
9. Corry, P., Kozan, E.: Ant colony optimisation for machine layout problems. Computational Optimization and Applications 28(3), 287–310 (2004)
10. Deneubourg, J.L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., Chrétien, L.: The Dynamics of Collective Sorting: Robot-like Ants and Ant-like Robots. In: From Animals to Animats, 1st International Conference on Simulation of Adaptive Behaviour, pp. 356–363 (1990)
11. Dorigo, M.: Optimization, Learning and Natural Algorithms. PhD thesis, Politecnico di Milano, Italie (1992)
12. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics B 26(1), 29–41 (1996)
13. Dorigo, M., Gambardelaa, L.M.: Ant Colony System:A cooperative Learning Approach to Travelling Salesman Problem. IEEE Trans. Evol. Comp. 1, 53–66 (1997)
14. Goldberger, A.L., Amaral, L.A.N., Glass, L., Hausdorff, J.M., Ivanov, P., Mark, R., Mietus, J., Moody, G., Peng, C., Stanley, H.: PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. Circulation 101(23), 215–220 (2000)
15. Grasse, P.: La reconstruction du nid et les coordinations inter-individuelles chez bellicositermes natalensis et cubitermes sp. La theorie de la stigmergie: Essai d'interperation des termites constructeurs. Insectes Sociaux 6, 41–81 (1959)
16. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Morgan Kaufmann Publishers, San Francisco (2006)
17. Hu, X., Zhang, J., Li, Y.: Orthogonal methods based ant colony search for solving continuous optimization problems. Journal of Computer Science and Technology 23(1), 2–18 (2008)
18. Jiang, W., Xu, Y., Xu, Y.: A novel data mining method based on ant colony algorithm. In: Li, X., Wang, S., Dong, Z.Y. (eds.) ADMA 2005. LNCS (LNAI), vol. 3584, pp. 284–291. Springer, Heidelberg (2005)
19. Jin, P., Zhu, Y., Hu, K., Li, S.: Classification Rule Mining Based on Ant Colony Optimization Algorithm. In: International Conference on Intelligent Computing (ICIC 2006). LNCIST, vol. 344, pp. 654–663. Springer, Heidelberg (2006)
20. Kantardzic, M., Zurada, J. (eds.): Next Generation of Data-Mining Applications. Wiley-IEEE Press, Chichester (2005)

21. Kargupta, H., et al.: Collective Data Mining. In: Karhgupta, Chan (eds.) Advances in Distributed Data Mining. MIT Press, Cambridge (2000)
22. Masaomi, K.: Application of Data Mining Techniques to the Data Analyses to Ensure Safety of Medicine Usage. In: Ponce, J., Karahoca, A. (eds.) Data Mining and Knowledge Discovery in Real Life Applications. I-Tech Education and Publishing (2009)
23. Kumar, S., Rao, C.: Application of ant colony, genetic algorithm and data mining-based techniques for scheduling. Robotics and Computer-Integrated Manufacturing 25, 901–908 (2009)
24. Kuo, R.J., Lin, S.Y., Shih, C.W.: Mining association rules through integration of clustering analysis and ant colony system for health insurance database in Taiwan. Expert Systems with Applications 33, 794–808 (2007)
25. Kuo, R.J., Wang, H.S., Hu, T.L., Chou, S.H.: Application of ant K-means on clustering analysis. Computers & Mathematics with Applications 50, 1709–1724 (2005)
26. Lioni, A., Sauwens, C., Theraulaz, G., Deneubourg, J.L.: Chain formation in Oecophylla longinoda. Journal of Insect Behavior 14, 679–696 (2001)
27. Liu, B., Abbass, H.A., McKay, B.: Classification rule discovery with ant colony optimization. IEEE Computational Intelligence Bulletin 3(1), 31–35 (2004)
28. Liu, H., Hussain, F., Tan, C.L., Dash, M.: Discretization: An enabling technique. Data Mining and Knowledge Discovery 6, 393–423 (2002)
29. Lopez-Ibanez, M., Blum, C.: Beam-ACO for the traveling salesman problem with time windows. Computers & Operations Research 37(9), 1570–1583 (2010)
30. Lumer, E.D., Faieta, B.: Diversity and Adaptation in Populations of Clustering Ants, From Animals to Animats. In: 3rd International Conference on the Simulation of Adaptive Behaviour, pp. 501–508 (1994)
31. Mhamdi, F., Elloumi, M.: A new survey on knowledge discovery and data mining. In: 2nd IEEE Int. Conf. on Research Challenges in Information Science, pp. 427–432 (2008)
32. Michelakos, I., Papageorgiou, E., Vasilakopoulos, M.: A Hybrid Classification Algorithm evaluated on Medical Data. In: 1st International Workshop on Cooperative Knowledge Discovery & Data Mining / 19th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (CKDD / WETICE), pp. 98–103 (2010)
33. Michelakos, I., Papageorgiou, E., Vasilakopoulos, M.: A Study of cAnt-Miner2 Parameters Using Medical Data Sets. In: 1st International Workshop on Cooperative Knowledge Discovery & Data Mining / 19th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (CKDD / WETICE), pp. 119–121 (2010)
34. Monmarche, N., Slimane, M., Venturini, G.: On improving clustering in numerical database with artificial ants. In: Floreano, D., Mondada, F. (eds.) ECAL 1999. LNCS (LNAI), vol. 1674, pp. 626–635. Springer, Heidelberg (1999)
35. Moss, J.D., Johnson, C.G.: An ant colony algorithm for multiple sequence alignment in bioinformatics. In: Pearson, D.W., Steele, N.C., Albrecht, R.F. (eds.) Artificial Neural Networks and Genetic Algorithms, pp. 182–186. Springer, Heidelberg (2003)
36. Mullen, R.J., Monekosso, D., Barman, S., Remagnino, P.: A review of ant algorithms. Expert Systems with Applications 36, 9608–9617 (2009)
37. Neumann, F., Sudholt, D., Witt, C.: Comparing variants of MMAS ACO algorithms on pseudo-boolean functions. In: Stützle, T., Birattari, M., Hoos, H.H. (eds.) SLS 2007. LNCS, vol. 4638, pp. 61–75. Springer, Heidelberg (2007)

38. Otero, F.E.B., Freitas, A.A., Johnson, C.G.: A Hierarchical Classification Ant Colony Algorithm for Predicting Gene Ontology Terms. In: Pizzuti, C., Ritchie, M.D., Giacobini, M. (eds.) EvoBIO 2009. LNCS, vol. 5483, pp. 339–357. Springer, Heidelberg (2009)
39. Otero, F.E.B., Freitas, A.A., Johnson, C.G.: cAnt-Miner: an ant colony classification algorithm to cope with continuous attributes. In: Dorigo, M., Birattari, M., Blum, C., Clerc, M., Stützle, T., Winfield, A.F.T. (eds.) ANTS 2008. LNCS, vol. 5217, pp. 48–59. Springer, Heidelberg (2008)
40. Otero, F.E.B., Freitas, A.A., Johnson, C.G.: Handling continuous attributes in ant colony classification algorithms. In: IEEE Symposium on Computational Intelligence in Data Mining (CIDM), pp. 225–231 (2009)
41. Parpinelli, R.S., Lopes, H.S., Freitas, A.A.: An ant colony based system for data mining: applications to medical data. In: Genetic and Evolutionary Computation Conference (GECC0 2001), pp. 791–797 (2001)
42. Parpinelli, R.S., Lopes, H.S., Freitas, A.A.: Data mining with an ant colony optimization algorithm. IEEE Transactions on Evolutionary Computation 6, 321–332 (2002)
43. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence 27, 1226–1238 (2005)
44. Pérez-Delgado, M.: Rank-Based Ant System to Solve the Undirected Rural Postman Problem. In: Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living, pp. 507–514 (2009)
45. Phokharatkul, P., Sankhuangaw, K., Somkuarnpanit, S., Phaiboon, S., Kimpan, C.: Off-Line Hand Written Thai Character Recognition using Ant-Miner Algorithm. Transactions on ENFORMATIKA on Systems Sciences and Engineering 8, 276–281 (2005)
46. Prather, J.C., Lobach, D.F., Goodwin, L.K., Hales, J.W., Hage, M.L., Hammond, W.E.: Medical Data Mining: Knowledge Discovery in a Clinical Data Warehouse. In: Annual Conference of the American Medical Informatics Association, pp. 101–105 (1997)
47. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)
48. Shmygelska, A., Aguirre-Hernández, R., Hoos, H.H.: An ant colony optimization algorithm for the 2D HP protein folding problem. In: Dorigo, M., Di Caro, G.A., Sampels, M. (eds.) ANTS 2002. LNCS, vol. 2463, pp. 40–52. Springer, Heidelberg (2002)
49. Shmygelska, A., Hoos, H.H.: An ant colony optimisation algorithm for the 2D and 3D hydrophobic polar protein folding problem. BioMed Central Bioinformatics 6(30) (2005)
50. Solnon, C.: Combining two pheromone structures for solving the car sequencing problem with Ant Colony Optimization. European Journal of Operational Research 191(3), 1043–1055 (2008)
51. Stützle, T., Hoos, H.H.: MAX MIN Ant System. Future Generation Computer Systems 16, 889–914 (2000)
52. Taniar, D. (ed.): Research and Trends in Data Mining Technologies and Applications. Idea Group Publishing, USA (2007)
53. Thangavel, K., Jaganathan, P.: Rule Mining Algorithm with a New Ant Colony Optimization Algorithm. In: International Conference on Computational Intelligence and Multimedia Applications, pp. 135–140 (2007)

54. Theraulaz, G., Bonabeau, E., Sauwens, C., Deneubourg, J.L., Lioni, A., Libert, F., Passera, L., Solé, R.: Model of droplet dynamics in the Argentine ant Linepithema humile (Mayr). Bulletin of Mathematical Biology 63, 1079–1093 (2001)
55. Tiwari, R., Husain, M., Gupta, S., Srivastava, A.: Improving ant colony optimization algorithm for data clustering. In: International Conference and Workshop on Emerging Trends in Technology, pp. 529–534 (2010)
56. Tsai, C.F., Tsai, C.W., Wu, H.C., Yang, T.: ACODF: a novel data clustering approach for data mining in large databases. Journal of Systems and Software 73(1), 133–145 (2004)
57. Wang, Z., Feng, B.: Classification rule mining with an improved ant colony algorithm. In: Webb, G.I., Yu, X. (eds.) AI 2004. LNCS (LNAI), vol. 3339, pp. 357–367. Springer, Heidelberg (2004)
58. White, T., Kaegi, S., Oda, T.: Revisiting elitism in ant colony optimization. In: Cantú-Paz, E., Foster, J.A., Deb, K., Davis, L., Roy, R., O'Reilly, U.-M., Beyer, H.-G., Kendall, G., Wilson, S.W., Harman, M., Wegener, J., Dasgupta, D., Potter, M.A., Schultz, A., Dowsland, K.A., Jonoska, N., Miller, J., Standish, R.K. (eds.) GECCO 2003. LNCS, vol. 2723, pp. 122–133. Springer, Heidelberg (2003)
59. Yang, J., Shi, X., Marchese, M., Liang, Y.: An ant colony optimization method for generalized TSP problem. Progress in Natural Science 18(11), 1417–1422 (2008)