Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 158632, 11 pages
http://dx.doi.org/10.1155/2015/158632

*Research Article*

# A Multiple Pheromone Table Based Ant Colony Optimization for Clustering

**Kai-Cheng Hu,[1] Chun-Wei Tsai,[2] Ming-Chao Chiang,[1] and Chu-Sing Yang[3]**

[1]*Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 80424, Taiwan*
[2]*Department of Computer Science and Information Engineering, National Ilan University, Yilan 26047, Taiwan*
[3]*Department of Electrical Engineering and Institute of Computer and Communication Engineering, National Cheng Kung University, Tainan 70101, Taiwan*

Correspondence should be addressed to Ming-Chao Chiang; mcchiang@cse.nsysu.edu.tw

Received 17 January 2015; Revised 13 May 2015; Accepted 17 May 2015

Academic Editor: Marek Lefik

Ant colony optimization (ACO) is an efficient heuristic algorithm for combinatorial optimization problems, such as clustering. Because the search strategy of ACO is similar to those of other well-known heuristics, the probability of searching particular regions will be increased if better results are found and kept. Although this kind of search strategy may find a better approximate solution, it also has a high probability of losing the potential search directions. To prevent the ACO from losing too many potential search directions at the early iterations, a novel pheromone updating strategy is presented in this paper. In addition to the "original" pheromone table used to keep track of the *promising* information, a second pheromone table is added to the proposed algorithm to keep track of the *unpromising* information so as to increase the probability of searching directions worse than the current solutions. Several well-known clustering datasets are used to evaluate the performance of the proposed method in this paper. The experimental results show that the proposed method can provide better results than ACO and other clustering algorithms in terms of quality.

## 1. Introduction

Partitional clustering is a classical NP-hard problem [1–3]. The goal of this problem is to divide a set of patterns into several different groups based on their features. Moreover, the basic concept of clustering is simply to put similar data together. For example, consider four flowers with different colors: red, yellow, blue, and purple. They can be split up into warm color group with red and yellow flowers and cool color group with blue and purple flowers. Clustering is important because technologies relevant to it can be applied to a large number of practical applications, such as bioinformation [4, 5], document analysis [6, 7], human face recognition [8, 9], and search engine [10–12].

Jain and Dubes [1] divided the clustering process into several steps: pattern representation, definition of a pattern proximity measure appropriate for the data domain, clustering or grouping, data abstraction, and assessment of output. Pattern representation refers to the number of data, groups, and other features available to the clustering algorithm. Pattern proximity uses objective function (or fitness function) to measure the difference of patterns. Two kinds of measurements are usually used for evaluating the results of clustering. The first kind considers the degree of closeness between patterns. For instance, one of the well-known measurements is to calculate the sum of distances between patterns and centroids to which the patterns belong. The second kind considers the difference among the groups. One widely used measurement is to compute the sum of distances among centroids. A large sum implies that groups are quite different from each other.

Since clustering technologies can be used in our daily life, a large number of search methods were presented to solve the partitional clustering problem of which $k$-means [13, 14] is one of the most widely used, for it is simple and easy to implement. Initially, $k$-means generates a set of centroids, the number of which is predefined. Then, the patterns will be assigned to the nearest groups (centroids). At this step, patterns assigned to the same group are regarded as in the same group. After the assignment process, $k$-means will then recalculate the centroid of each group by averaging the positions of patterns in that group. The assignment and update processes will be repeated until the positions of all centroids do not change or the changes are less than a predefined threshold. Although $k$-means are simple and easy to implement, the result is extremely sensitive to the initial solution. Another drawback is that $k$-means is easily falling into local optima, and there is simply no way to escape from the local optima.

Therefore, many researches attempt to solve the clustering problem using different metaheuristic algorithms. Tabu search (TS) [15] uses a tabu list to keep track of solutions that have been tried recently so as to avoid searching for the same solutions repeatedly in the near future. Genetic algorithm (GA) [16, 17] uses the crossover and mutation operations to share information available between chromosomes to enlarge the search region. Recently, another promising research trend has been using particle swarm optimization (PSO) [18, 19] that takes into account both the population experience and the individual experience to obtain a better clustering result than traditional clustering algorithms do. Another swarm intelligence, ant colony optimization (ACO) [20], has also been applied to the clustering problem, by having the ants put pheromone on the paths they passed through so as to share the information learned by each ant. This strategy can also lead the algorithm to find good solutions.

Generally speaking, ACO is a powerful search method, especially for combinatorial optimization problems, because all the search information can be shared and accumulated on the pheromone table. However, for some optimization problems, ACO also suffered from the problem of falling into local optima at early iterations even though it has a better chance to select different search directions. Also, paths with higher pheromone concentration do not always lead to the final goal. Hence, we present in this paper an ACO-based clustering algorithm, called multiple pheromone table based ant colony optimization (MPTACO), to solve this problem (this is basically an extended version of our previous study [21]). A second pheromone table is added to the ACO to keep track of the unpromising information so that the ACO can try to find better clustering results by using this unpromising information. To evaluate its performance, the proposed method is compared with four state-of-the-art algorithms: standard $k$-means [14], genetic $k$-means algorithm [16], ant colony system (ACS) [22], and ACODPT [21].

The remainder of the paper is organized as follows. Section 2 gives a brief introduction to the ACO-based algorithms and improvement on them. Section 3 describes in detail the proposed algorithm. Performance evaluation of the proposed algorithm is presented in Section 4, and the conclusion is drawn in Section 5.

## 2. Related Work

*2.1. Ant Colony Optimization.* The basic idea of ant colony optimization (ACO) comes from the foraging behavior of ant colony [23]. The foraging behavior of an ant colony is that ants will leave pheromone on the paths they passed through for the other ants to find the food. The ants tend to select the path with higher pheromone concentration. For this reason, a path with higher pheromone concentration will lead more ants to go through the path, implying that the path has a larger chance to find food.

```
(1) ACO()
(2) {
(3)   Initialization()
(4)   **While** the termination criterion is not met
(5)     Solution_Construction()
(6)     Pheromone_Upuate()
(7)     Local_Search()
(8)   **End**
(9) }
```

Algorithm 1: Outline of the ACO algorithm.

Algorithm 1 gives an outline of the ACO algorithm. In the solution construction process, the ants need to decide which paths to take. To realize this concept, the probability of choosing $(r, s)$ as the next path is defined as follows:

$$p_k(r,s) = \begin{cases} \dfrac{[\tau(r,s)]^{\alpha} \cdot [\eta(r,s)]^{\beta}}{\sum_{u \in J_k(r)} [\tau(r,u)]^{\alpha} \cdot [\eta(r,u)]^{\beta}} & \text{if } s \in J_k(r), \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $J_k(r)$ denotes the set of candidates which ant $k$ can reach from the current position $r$; $\tau(r,s)$ is the pheromone value on the path $(r,s)$; $\eta(r,s)$ is the inverse of the distance of the path $(r,s)$; $\alpha$ and $\beta$ are the weight of exploration and exploitation, respectively. The operator for updating the pheromone concentration $\tau(r,s)$ on each path $(r,s)$ is defined as follows:

$$\begin{aligned} \tau(r,s) &= (1-\rho)\cdot\tau(r,s) + \rho\sum_{k=1}^{m}\Delta\tau_{(r,s)}^{k}, \\ \Delta\tau_{(r,s)}^{k} &= (L_k)^{-1}, \end{aligned} \tag{2}$$

where $\rho$ denotes the pheromone decay which is a value in the range $(0, 1)$; $m$ is the number of ants; $L_k$ is the length of the tour created by ant $k$.

In a later research [22], Dorigo et al. presented another algorithm called ant colony system (ACS) which is an extended version of [23]. ACS enhances the convergence of the original ACO. The algorithm gives ants a certain chance to directly select the path with the highest pheromone concentration as follows:

$$s = \begin{cases} \arg\max\limits_{u \in J_k(r)} \left\{[\tau(r,u)] \cdot [\eta(r,u)]^{\beta}\right\} & \text{if } q \le q_0, \\ \mathcal{S} & \text{otherwise,} \end{cases} \tag{3}$$

where $q$ denotes a random value and $q_0$ is a threshold for determining the strategy to be used.

The first strategy selects the best next partial solution in the case $q \le q_0$ while the second strategy selects the next partial solution $\mathcal{S}$ determined by the following probability distribution:

$$p_k(r,s) = \begin{cases} \dfrac{[\tau(r,s)] \cdot [\eta(r,s)]^{\beta}}{\sum_{u \in J_k(r)} [\tau(r,u)] \cdot [\eta(r,u)]^{\beta}} & \text{if } s \in J_k(r), \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

For this reason, the ant will not always check the probability of each path. Therefore, ACS can speed up the convergence process, thus reducing some of the computation time.

*2.2. Ant Colony Optimization for Clustering and Improvements.* Shelokar et al. [24] presented an ACO-based clustering algorithm which uses pheromone trails. The main idea is transferring the meaning of information in the solutions. Different from the original version of ACO for the traveling salesman problem (TSP) in which the solutions record the order of the cities the salesman travels, for the clustering problem, the solutions [24] record the cluster numbers to which the patterns are assigned, by using an $N$ by $K$ pheromone matrix where $N$ is the number of nodes and $K$ is the number of centroids. The matrix records the pheromone concentration on the path between each pattern and each centroid. As illustrated in Figure 1, unlike the pheromone table which is represented by a matrix, each solution is encoded as a one-dimensional array, the indices of which correspond to the pattern number while the value of each element corresponds to the cluster to which that pattern is assigned. The example shows that pattern 1 is assigned to centroid 3, pattern 2 is assigned to centroid 5, pattern 3 is assigned to centroid 8, and so on. By keeping track of to which cluster each pattern is assigned as illustrated in Figure 1, ACO can be applied to the clustering problem. Moreover, the local search method is also added to speed up the convergence.

In [25], different strategies are added to the ACO to decide the solution. One of the strategies is that ants only check part of the paths instead of all the paths. Another strategy is using a threshold of pheromone to split patterns into different groups and checking the groups at the end. If only a few patterns are in a group and the centroid of that group is close to the centroid of another group, the two groups are merged. In [26], several additional steps are used in the ACO. First, the root cluster, which contains all the patterns, is split into several smaller clusters, and each pattern is assigned to a suitable cluster. Then, clusters which are close to each other are merged. Finally, dissimilar patterns are removed from their clusters, and new clusters for these patterns are constructed.

More recently, Tiwari et al. [27] used two strategies to enhance the performance of ACO. One is setting the current value of pheromone to the initial value once every 50 iterations while the other is setting the pheromones on all paths to the initial value if the pheromones on the paths remain intact for the past 10 iterations in a row. Akarsu and Karahoca [28] used sequential backward selection (SBS) to reduce the dimensions of patterns. They use Manhattan distance as the fitness function and consider the pheromone only when the ants select paths. Jiang et al. [29] split the structure of solutions into two parts. The first part is used to decide which patterns will be included in the following computation. The second part is used to record the relationship between patterns and clusters. Some researches are focused on self-adaptive clustering. For instance, Liu and Fu [30] presented a method which uses Jaccard index to decide a suitable cluster number. Niknam and Amiri [31] presented a method which combines fuzzy adaptive particle swarm optimization (FAPSO), ACO, and $k$-means algorithm. The experiment results show that these methods get a better performance.

| 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|
| 3 | 5 | 8 | 1 | 3 | 1 |

Figure 1: How solutions are encoded.

*2.3. Local Minima Problem of Clustering Algorithm.* Since the clustering problem is a traditional optimization problem, a large number of clustering algorithms have been presented for several years [1]. As we mentioned in Section 1, $k$-means is one of the most well-known clustering algorithms, but its search process may fall into local minima because it is very sensitive to the randomly generated initial solution and it does not have any mechanism to escape the local optima [32]. Metaheuristic algorithms provide an alternative way to prevent the search process from falling into local minima. One of them is the simulated annealing [33] which has a small chance to accept a solution that is worse than the current solution as the new search direction. In [34], tabu search used the so-called tabu list to avoid the search process from falling into local minima. Different from simulated annealing and tabu search, genetic algorithm (GA) [35] used multiple search directions to keep the search direction from falling into local minima or searching the same region repeatedly. Also because GA has a mutation operator, a small percentage of subsolutions can be disturbed. This implies that GA has a chance to escape the local minima. Particle swarm optimization [36] and ant colony optimization [23] also used multiple search directions to search for the solution; they will not easily fall into local minima at early iterations. In summary, most clustering algorithms will confront the dilemma of local minima; therefore, how to mitigate this problem has become an active research topic in recent years. Using the metaheuristic algorithm alone and combining it with traditional clustering algorithms are two promising solutions to keep the search process from falling into local minima at early iterations [3].

## 3. The Proposed Method

In this section, we will describe in detail the proposed algorithm MPTACO in three steps: first the concept, then the proposed algorithm, and finally an example.

*3.1. Concept.* In general, most ACO-based algorithms, even most metaheuristic algorithms, focus on the positive information. For example, for the ACO, ants move along the trail with higher pheromone; for GA, fitter chromosomes survive; for PSO, particles move toward the best particle. However, searching the so-called good directions does not always lead to good results. In fact, the trajectory of search is tortuous when dealing with complex problems. To prevent the ACO from choosing only paths that have high pheromone values and remove paths that have low pheromone values so as to increase the search diversity on the convergence process, the proposed algorithm adds a second pheromone table to keep track of negative information. The consequence is that the proposed algorithm has a better opportunity to select worse partial solutions during the search process so as to find solutions better than those the original ACO can find. As we mentioned in Section 2.3, although most metaheuristic algorithms have mechanisms to avoid their search process from falling into local minima at early iterations, they do not guarantee that their search process will not fall into local minima. Because the search direction of metaheuristic algorithms typically is toward a result that is better in terms of the fitness value (i.e., the value returned by a fitness or objective function), if the algorithm cannot find a better result for a long while before it finds the global optima, the search process might get stuck at a particular position or region. This phenomenon is called falling into local optima. Most of the metaheuristic algorithms have some strategies to escape the local optima.

According to our observation, the search process of ACO will move toward particular solutions (or paths), because the pheromone values associated with these solutions will be strengthened if they are better than the other solutions ACO finds. One of the reasons is that ACO keeps only the promising solutions. To avoid the search process from moving toward particular search directions, a different way to construct the solutions is considered in this paper, that is, taking into account the information of unpromising solutions ACO finds. As shown in Figure 2, $R_i$ denotes the $i$th pattern, $C_j$ is the $j$th centroid, and a solid line is the assignment of a pattern to a centroid. This example shows that pattern $R_1$ is assigned to centroid $C_3$, pattern $R_2$ is assigned to centroid $C_1$, and pattern $R_3$ is assigned to centroid $C_2$. The ant picks first path $(R_1, C_3)$ and then path $(R_2, C_1)$. Afterwards, the proposed algorithm eliminates the path $(R_3, C_3)$, based on values in the negative pheromone table and the selection strategy, and then selects path $(R_3, C_2)$. The dashed line denotes the virtual path, which is used to show that the ant moves continuously. In the following subsection, we will discuss briefly the concept first and then explain how the ants choose paths.

*3.2. Algorithm.* As shown in Figure 3, the major difference between the proposed method and the ACO is in that MPTACO employs two tables, which we refer to as the positive pheromone table and the negative pheromone table, in the transition phase to decide which path an ant is supposed to take. The positive pheromone table is the only pheromone table in ACO, which is used to select paths in the candidate list, whereas the negative pheromone table is used by MPTACO to eliminate paths from the candidate list.

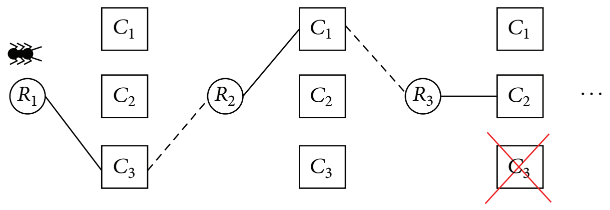

Figure 2: An illustration of MPTACO.

*3.2.1. Initialization.* In the initialization phase, that is, initially, the value of each path in the positive pheromone table is set equal to $\tau_0$ while the value of each path in the negative pheromone is set equal to 1. Moreover, since the negative pheromone table is used to keep track of the probability of selection, all the paths will be in the candidate list at the very beginning. For each ant, the initial partial solution is set randomly. In other words, each pattern is assigned to a cluster randomly, and then the centroids are calculated. The proposed method represents solutions in such a way that it is the same as described in [24], that is, to which centroid each pattern belongs. Instead of binary encoding, integer encoding is used for two reasons: (1) it is easier to understand the meaning of the solutions, and (2) it is shorter to integer encode the solutions. But, on the other side, integer encoding usually takes more memory space than binary encoding in storing the information.

*3.2.2. Transition.* This phase decides the paths in which ants move. In other words, this phase decides the relationship between patterns and centroids. Two kinds of strategies are used for the transition of the proposed algorithm. Equation (5) shows how the decision is made in the proposed algorithm for the transition phase:

$$s = \begin{cases} \arg\max\limits_{u \in N_k(r)} \left\{ [\tau(r,u)] \cdot [\eta(r,u)]^{\beta} \right\} & \text{if } q \le C_r, \\ V & \text{otherwise,} \end{cases} \quad (5)$$

where $C_r$ is the number used to decide which strategy to choose, $\tau(r,s)$ and $\eta(r,s)$ are the same as defined in ACS, $\beta$ is the weight to control the influence of distance, and $V$ is a random variable that is used to select the next partial solution based on the following probability distribution:

$$q^k(r,z) = \begin{cases} \dfrac{[\tau(r,z)] \cdot [\eta(r,z)]^{\beta}}{\sum_{u \in N_r^k} [\tau(r,u)] \cdot [\eta(r,u)]^{\beta}} & \text{if } z \in N_r^k \setminus B_r^k, \\ 0 & \text{otherwise,} \end{cases} \quad (6)$$

with $q^k(r,z)$ representing the probability of pattern $r$ of ant $k$ belonging to centroid $z$ in the next partial solution, $N_r^k$ the set of paths which ant $k$ can reach from pattern $r$, and $B_r^k$, generated by (7), the set of paths which will not be selected:

$$B_r^k = B_r^k \cup \{\overline{rz}\} \quad \forall z \in N_r^k,\ \tau^b(r,z) < q_b, \quad (7)$$

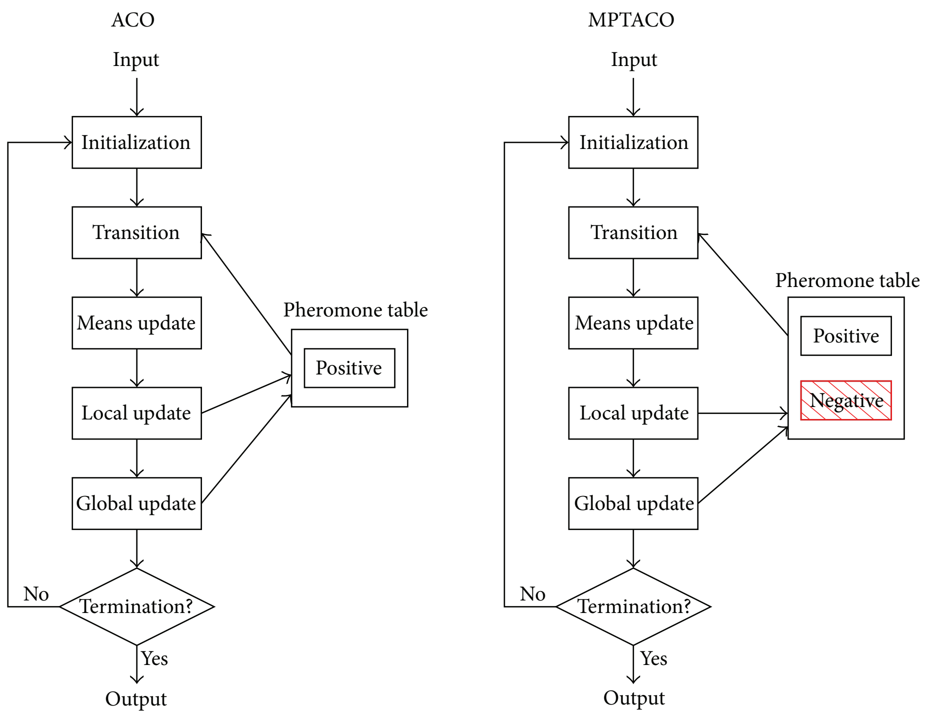

Figure 3: The difference between ACO and MPTACO.

where $q_b$ is a random number uniformly distributed in the range of 0 to 1. It will be regenerated every time the algorithm checks the negative pheromone.

Like ACS, the ants have some chances to select the path with the highest positive pheromone concentration. The negative pheromone used in the second strategy is the probability to determine whether a path will be added to the candidate list or not. Then, the ants will select the paths in terms of the positive pheromone from the candidate list. This implies that the proposed algorithm has an opportunity to eliminate the paths which are really bad. The remaining paths will be chosen with a higher probability. Moreover, the main difference between MPTACO and ACODPT [21] is in the transition phases of them. For ACODPT, three search strategies are used by each ant. For MPTACO, the search strategies for each ant have been simplified to two, and $C_r$, which is defined as

$$C_r = 0.5 \times \frac{\text{current iteration number}}{\text{total number of iterations}}, \quad (8)$$

is used to increase the search diversity at early iterations of the proposed algorithm.

Figure 4 gives a simple example to illustrate how the transition phase works. First, the example shows that a number of paths can be selected, each of which is associated with a pair of numbers denoted by $x/y$ where $x$ denotes the positive pheromone and $y$ is the negative pheromone. For instance, the positive pheromone of path 1 is 10.0 while the negative pheromone is 1.0. The first strategy will select the path with the most positive pheromone, so path 1 is selected. The second strategy assumes that path 3 has been eliminated so that

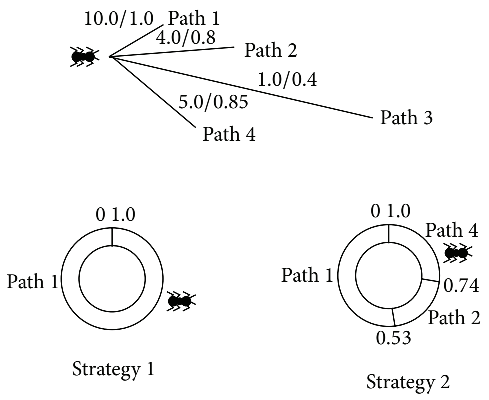

Figure 4: Example for illustrating the transition.

the probability of choosing the remaining paths increases. Here we assume that path 4 is selected in this example.

In general, the proposed algorithm will search more broadly at early iterations and more deeply at later iterations. For this reason, the proposed algorithm has a larger chance to select the first strategy at early iterations and the second strategy at later iterations. Which strategy is taken depends on the threshold $C_r$ defined above.

*3.2.3. Means Update.* The proposed algorithm will then update the positions of centroids by using the current partial solution for each ant as follows:

$$\frac{\sum_{j=1}^{n_i} x_{ij}}{n_i}, \quad (9)$$

where $x_{ij}$ denotes the $j$th pattern of $i$th centroid and $n_i$ is the number of patterns belonging to centroid $i$.

*3.2.4. Local Pheromone Update.* Same as described in [23], the positive pheromone table will be updated by using the following equation:

$$\tau(r,s) = (1-\rho)\cdot\tau(r,s) + \rho\cdot\tau_0, \tag{10}$$

where $\tau(r,s)$ denotes the pheromone value associated with the path $(r,s)$, $\rho$ is the evaporation degree, and $\tau_0$ is the initial pheromone on each path. This phase updates the positive pheromone on the paths which are taken by the ants. If a path is taken by more ants, the positive pheromone on the path will be higher.

*3.2.5. Global Pheromone Update.* This phase will update both the positive and the negative pheromone tables. The algorithm will choose the best ant so far to update the positive pheromone table. The positive pheromone on the paths passed through by the best ant will be enhanced. The best ant means the ant with the best fitness value. For example, if the sum of squared errors (SSE) is used as the fitness value, the ant with the smallest SSE will be the best. The pheromone update on all the paths $(r,s)$ of the best ant is defined as

$$\tau(r,s) = (1-\alpha)\cdot\tau(r,s) + \alpha L_{\text{best}}^{-1}, \tag{11}$$

where $\alpha$ denotes the influence degree of the best ant and $L_{\text{best}}$ is the total distance which the best ant has moved. As for the updating of negative pheromone table, the proposed algorithm will select the worst ant so far to update the negative pheromone on the paths through which it passed. Same as the best ant, the worst ant means the ant with the worst fitness value. For every path $(r,s)$ the worst ant has passed through, the negative pheromone will be updated as follows:

$$\tau^b(r,s) = \gamma\tau^b(r,s), \tag{12}$$

where $\gamma$ is a constant in the range $[0,1]$, indicating the evaporation rate of negative pheromone. Besides, the proposed algorithm will also update the negative pheromone on all the paths $(r,s)$ which are passed through by the best ant so far as follows:

$$\tau^b(r,s) = \min\left\{\tau^b(r,s)\cdot\gamma^{-1}, 1\right\}. \tag{13}$$

Because $\tau^b(r,s)$ is used as a probability to decide whether the path $(r,s)$ should be put into the candidate list or not, the value of $\tau^b(r,s)$ has to be in the range $[0,1]$. That is, if $\tau^b(r,s)$ is larger than 1, it will be set to 1. A larger negative pheromone of the path $(r,s)$ means that the path $(r,s)$ has a higher probability of being put into the candidate list. The path passed through by the best ant means that it may be the best choice of the final solution. So the selection probability of the path will increase.

*3.2.6. Termination Criterion.* The termination criteria can vary. For instance, the search process can be terminated if the total distance of any ant is shorter than a threshold or if the difference between ants is less than a threshold. In this paper, the search process will be terminated if the number of iterations has reached a predefined number.

*3.3. Summary.* In ACO, only one kind of pheromone is used. All the ants use the same kind of pheromone to decide the search directions. However, using only one kind of pheromone to decide the search directions may neglect some search directions which have the potential to lead to better results, and it may also lead the search process to fall into local optima easily. For this reason, ACODPT adds the second kind of pheromone, which is referred to as the negative pheromone, to improve the quality of the end result. The negative pheromone is used to eliminate the search directions, thus increasing the chance the remaining search directions are selected. This way, the proposed method can have a higher probability to search for all the potential directions. Compared to ACODPT and MPTACO, the number of decision strategies in the transition operator of ACODPT is three, but it has been reduced to two in MPTACO. More important, the transition operator of MPTACO can select the decision strategies automatically according to the search time instead of the probability.

Comparing to the hill-climbing method, ACO, ACODPT, and MPTACO do not always search for better solutions. The selection strategies give them chances to select worse directions. If the algorithm always accepts a better solution just like the hill-climbing method does, it will fall into local optima at early iterations because it does not search for other possibilities in a large search space. Moreover, the pheromone tables and the corresponding strategies make the proposed method capable of mitigating the problem of falling into local optima quickly.

## 4. Experimental Results

*4.1. Experimental Environment and Datasets.* The experiments are conducted on a PC with 2.67 GHz Intel Core i7-920 CPU and 4 GB of memory running Fedora 12 with Linux 2.6.31.5-127. Also, the programs are written in C and compiled by gcc version 4.4.2 20091027 (Red Hat 4.4.2-7).

The datasets used in the experiments are taken from UCI [37], the details of which are as described in Table 1.

*4.2. Simulation Results.* To evaluate the performance of MPTACO, we compare it with four state-of-the-art clustering algorithms: standard $k$-means [14], genetic $k$-means algorithm (GKA) [16], ant colony system (ACS) [22], and ACODPT [21]. The reasons that these algorithms are chosen for the purpose of comparison are in order. Standard $k$-means is a simple, easy to implement, and widely used algorithm for the clustering problem. It is a single-solution-based algorithm and has the characteristic of not being able to escape the local optima. For this reason, $k$-means can be used to compare the difference when using strategies that can escape the local optima. GKA for clustering leverages the strength of genetic algorithm (GA) and $k$-means; thus, it is not only capable of

Table 1: The datasets.

| | Number of patterns | Number of clusters | Number of attributes |
|---|---|---|---|
| Abalone | 4177 | 29 | 7 |
| Breast Cancer Wisconsin (Diagnostic) | 569 | 2 | 30 |
| Connectionist Bench (Sonar, Mines versus Rocks) | 208 | 2 | 60 |
| Ecoli | 336 | 8 | 7 |
| Glass Identification | 214 | 7 | 9 |
| Haberman's Survival | 306 | 2 | 3 |
| Iris | 150 | 3 | 4 |
| Parkinsons | 195 | 2 | 22 |
| Wine | 178 | 3 | 13 |
| Yeast | 1484 | 10 | 8 |
| Zoo | 101 | 7 | 16 |

Table 2: Quality.

| | $k$-means | GKA | ACS | ACODPT | MPTACO |
|---|---|---|---|---|---|
| Abalone | 28.15 ± 0.72 | 27.75 ± 0.41 | 26.61 ± 0.20 | 26.52 ± 0.29 | **26.44 ± 0.31** |
| Breast Cancer Wisconsin (Diagnostic) | 1190220.00 ± 0.00 | 1157240.00 ± 21592.60 | 1143100.00 ± 0.00 | 1143100.00 ± 0.00 | **1143100.00 ± 0.00** |
| Connectionist Bench (Sonar, Mines versus Rocks) | 281.63 ± 5.63 | 280.54 ± 0.01 | 280.55 ± 0.01 | 280.53 ± 0.00 | **280.53 ± 0.00** |
| Ecoli | 2284.85 ± 185.76 | 1816.42 ± 0.00 | 1834.75 ± 34.47 | 1939.17 ± 179.95 | **1816.00 ± 19.25** |
| Glass Identification | 336.14 ± 22.51 | 298.99 ± 10.48 | 292.25 ± 0.00 | 292.25 ± 0.00 | **292.25 ± 0.00** |
| Haberman's Survival | 30524.40 ± 22.99 | 30507.00 ± 0.00 | 30508.20 ± 0.78 | 30507.00 ± 0.00 | **30507.00 ± 0.00** |
| Iris | 85.34 ± 19.17 | 78.94 ± 0.00 | 78.94 ± 0.00 | 78.94 ± 0.00 | **78.94 ± 0.00** |
| Parkinsons | 1344370.00 ± 5760.48 | 1343320.00 ± 77.84 | 1343100.00 ± 0.02 | 1343100 ± 0.02 | **1343100.00 ± 0.02** |
| Wine | 2370690.00 ± 0.00 | 2370690.00 ± 0.00 | 2370690.00 ± 0.00 | 2370690.00 ± 0.00 | **2370690.00 ± 0.00** |
| Yeast | 49.06 ± 3.08 | **45.76 ± 0.35** | 46.32 ± 1.20 | 46.28 ± 1.14 | 45.93 ± 1.17 |
| Zoo | 1185.76 ± 64.54 | 1076.06 ± 12.58 | 1057.50 ± 0.00 | **1057.50 ± 0.00** | 1057.72 ± 1.21 |

a deep search, but also capable of escaping the local optima. Because ACODPT is based on ACS and MPTACO is based on ACODPT, this explains why ACS and ACODPT are used to evaluate the performance of MPTACO.

The parameter settings of each algorithm are as follows. Except $k$-means, the population sizes of GKA, ACS, and MPTACO are fixed at 30. For the ACS and MPTACO, the population size is the number of ants. For the GKA, the crossover operator is replaced by a 2-iteration $k$-means operator, the mutation probability is 0.01, and the number of generations is 1,000. For the ACS and MPTACO, $\alpha$ is set to 0.1, $\beta$ to 11.0, $\rho$ to 0.1, and $\tau$ to 0.01. For the ACS, $q_0$ is set to 1. All the algorithms are run for 30 times, and the averages are taken as the experimental results. Tables 2 and 3 show, respectively, the quality and the computation time. The numbers in bold indicate which algorithm gives the best result for the dataset evaluated. The quality is measured by the sum of squared errors (SSE) defined as

$$\mathrm{SSE} = \sum_{i=1}^{k}\sum_{j=1}^{n_i}\left\|x_{ij} - c_i\right\|^2, \qquad (14)$$

and the numbers after ± are the standard deviation. This implies that the smaller the value, the better the result.

Table 2 shows that the proposed method outperforms the other algorithms evaluated in most cases in terms of the quality. $k$-means gets the worst results for all datasets because it has no strategy to escape the local optima. As for GKA, it combines GA and $k$-means to get rid of the weakness of $k$-means. This is why GKA can find a better result, even the best result for some of the datasets. ACS has a strong global search capability, as is reflected in complex datasets like abalone. Nevertheless, the local search capability of ACS is not as powerful as that of GKA. As such, for some simple datasets, the result is worse than that of GKA. The proposed method beats ACS not only for complex datasets but also for simple datasets. For some datasets, the proposed method provides a worse result than that of GKA and ACODPT. This can be easily justified by observing that the proposed method lacks the local search strategy to fine-tune the result.

Table 3 shows the computation time of each algorithm. Undoubtedly, $k$-means takes the least time because it only searches toward the best direction. The performance of the proposed method is the worst in terms of the computation

Table 3: Computation time.

| | $k$-means | GKA | ACS | ACODPT | MPTACO |
|---|---|---|---|---|---|
| Abalone | **11.95 ± 0.07** | 121.26 ± 1.51 | 772.96 ± 1.74 | 813.11 ± 1.26 | 810.63 ± 2.36 |
| Breast Cancer Wisconsin (Diagnostic) | **0.02 ± 0.00** | 0.42 ± 0.02 | 0.94 ± 0.00 | 0.98 ± 0.00 | 1.00 ± 0.01 |
| Connectionist Bench (Sonar, Mines versus Rocks) | **0.38 ± 0.01** | 4.10 ± 0.01 | 18.97 ± 0.01 | 19.76 ± 0.02 | 19.97 ± 0.03 |
| Ecoli | **0.30 ± 0.02** | 2.80 ± 0.15 | 18.88 ± 0.08 | 19.62 ± 0.06 | 20.23 ± 0.19 |
| Glass Identification | **0.21 ± 0.01** | 2.17 ± 0.10 | 12.11 ± 0.05 | 12.64 ± 0.06 | 12.91 ± 0.11 |
| Haberman's Survival | **0.04 ± 0.00** | 0.96 ± 0.01 | 4.43 ± 0.02 | 4.64 ± 0.02 | 4.92 ± 0.03 |
| Iris | **0.04 ± 0.00** | 0.70 ± 0.01 | 3.21 ± 0.01 | 3.40 ± 0.01 | 3.58 ± 0.02 |
| Parkinsons | **0.14 ± 0.00** | 1.82 ± 0.01 | 7.71 ± 0.01 | 8.09 ± 0.02 | 8.30 ± 0.03 |
| Wine | **0.11 ± 0.00** | 1.51 ± 0.01 | 6.40 ± 0.02 | 6.64 ± 0.02 | 6.87 ± 0.02 |
| Yeast | **1.86 ± 0.02** | 20.77 ± 0.33 | 107.08 ± 0.24 | 111.81 ± 0.17 | 113.60 ± 0.47 |
| Zoo | **0.16 ± 0.00** | 1.68 ± 0.05 | 14.09 ± 3.71 | 27.97 ± 5.50 | 29.84 ± 5.88 |

Table 4: Quality for normalized datasets.

| | $k$-means | GKA | ACS | ACODPT | MPTACO |
|---|---|---|---|---|---|
| Abalone | 21.94 ± 0.51 | 21.52 ± 0.37 | 21.25 ± 0.20 | **21.15 ± 0.16** | 21.16 ± 0.19 |
| Breast Cancer Wisconsin (Diagnostic) | 26.04 ± 1.04 | 24.96 ± 0.24 | 24.86 ± 0.00 | 24.86 ± 0.00 | **24.86 ± 0.00** |
| Connectionist Bench (Sonar, Mines versus Rocks) | 448.71 ± 6.03 | 445.56 ± 0.02 | 445.64 ± 0.03 | 445.53 ± 0.00 | **445.53 ± 0.00** |
| Ecoli | 6.92 ± 0.75 | 5.50 ± 0.41 | 5.19 ± 0.16 | 5.19 ± 0.16 | **5.16 ± 0.01** |
| Glass Identification | 18.35 ± 1.34 | 16.38 ± 0.28 | 16.20 ± 0.00 | 16.20 ± 0.00 | **16.20 ± 0.00** |
| Haberman's Survival | 25.33 ± 0.05 | 25.28 ± 0.00 | 25.28 ± 0.00 | 25.28 ± 0.00 | **25.28 ± 0.00** |
| Iris | 7.59 ± 1.30 | 7.00 ± 0.00 | 7.00 ± 0.00 | 7.00 ± 0.00 | **7.00 ± 0.00** |
| Parkinsons | 98.68 ± 0.00 | 98.68 ± 0.00 | 98.68 ± 0.00 | 98.68 ± 0.00 | **98.68 ± 0.00** |
| Wine | 48.98 ± 0.02 | 48.95 ± 0.00 | 48.95 ± 0.00 | 48.95 ± 0.00 | **48.95 ± 0.00** |
| Yeast | 65.96 ± 5.68 | **58.45 ± 0.25** | 59.67 ± 2.78 | 59.25 ± 2.26 | 58.46 ± 0.18 |
| Zoo | 34.02 ± 1.38 | 30.72 ± 0.31 | 30.51 ± 0.00 | 30.51 ± 0.00 | **30.51 ± 0.00** |

time. This is basically a trade-off between quality and computation time. For applications the computation time of which is not at a premium, but the quality is a major concern, then the proposed algorithm is a good choice.

Besides the previous experiments, we are also interested in the scales of the values in the datasets. It is obvious that attributes of datasets may influence the clustering result because of the different scales of the values in the datasets. Table 4 shows the results after the attributes of all the datasets are normalized to be in the range [0, 1]. As can be easily seen from the results, the best value of some datasets has been changed. This implies that the scale of attributes can influence the clustering result.

*4.3. Impact of Parameters.* In this experiment, we test the influence of the parameter $\beta$, which is used to control the proportion between positive pheromone and distance. Figure 5 shows the results. The four datasets tested are abalone, yeast, normalized abalone, and normalized yeast. The table shows that a larger $\beta$ implies a better quality. This means that the distance has a stronger influence on the result than the positive pheromone. This is because SSE is used as the fitness function, which considers only distances between patterns

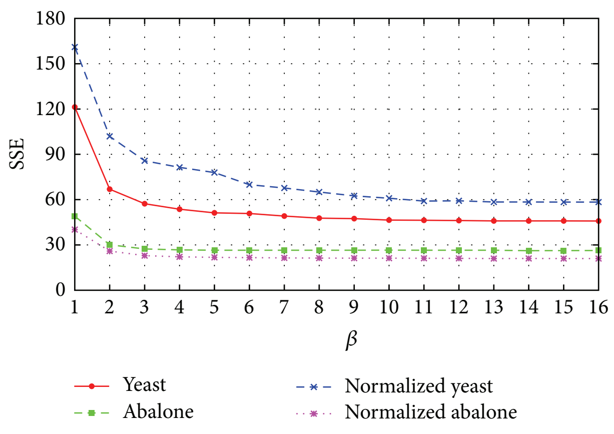

Figure 5: The experiment result of $\beta$.

and centroids. If a different fitness function is used, the result may be different.

$C_r$ in (8) is used to automatically decide the search strategy according to the total number of iterations and the current iteration number. At the beginning of the search procedure, the diversification is respected to discover more regions

```
(1) MPTACO()
(2) {
(3)     // Initialization
(4)     For each pair of pattern $r$ and centroid $s$ {
(5)         $\tau(r, s) = \tau_0$
(6)         $\tau^b(r, s) = 1$
(7)     }
(8)     While (the termination criterion is not met) {
(9)         // Transition
(10)        For each ant $k$
(11)            For each pattern $r$
(12)                Assign $r$ to centroid $s$ using (5), (6), and (7)
(13)        // Means update
(14)        For each ant $k$
(15)            For each centroid $s$
(16)                Update the position of $s$ using (9)
(17)        // Local pheromone update
(18)        For each ant $k$
(19)            For each pair of pattern $r$ and centroid $s$
(20)                If $(r, s)$ is selected by ant $k$
(21)                    Update $\tau(r, s)$ using (10)
(22)        // Golbal pheromone update
(23)        For each pair of pattern $r$ and centroid $s$ {
(24)            If $(r, s)$ is selected by the best ant
(25)                Update $\tau(r, s)$ and $\tau^b(r, s)$ using (11) and (13)
(26)            If $(r, s)$ is selected by the worst ant
(27)                Update $\tau^b(r, s)$ using (12)
(28)        }
(29)    }
(30)    Output the best result
(31) }
```

Pseudocode 1: Pseudocode of the proposed algorithm MPTACO.

that may have better local optima. The algorithm will have a higher chance to choose the search direction which is not the best at the moment. As the time passes by, the search procedure will focus on the intensification. In the end of the search procedure, the algorithm tries to converge to the local optima of the regions. In other words, this strategy makes the algorithm search widely at the beginning but deeply in the end.

## 5. Conclusions and Future Works

In this paper, we proposed an efficient algorithm, called MPTACO, to enhance the quality of the ACO-based algorithms for clustering. The idea is adding a second table to record the negative pheromone and to eliminate the most impossible paths so as to increase the probability of choosing worse paths in the table. It can prevent the ants from falling into local optima again and again. In addition, MPTACO can automatically adjust the convergence process according to the number of iterations carried out so far. The experimental results show that the proposed method can get a better result in most cases, especially for large-scale and complex datasets. Our future goal is to reduce the computation time of the proposed method while at the same time enhancing its quality. Also, another concern is how to use the negative pheromone table more efficiently.

## Appendix

## Pseudocode of MPTACO

Pseudocode 1 gives the pseudocode of the proposed algorithm MPTACO. As the pseudocode shows, the positive and negative pheromones of each path will be initialized to $\tau_0$ and 1, respectively, before the main loop is performed repeatedly until the termination criterion is met, as follows. First, all the ants are transited to the new states in the transition phase, as lines (10)–(12) show. Next, in the means update phase, MPTACO will update the centroids of each ant, as lines (14)–(16) show. Then, in the local pheromone update phase, the positive pheromone values will be updated, as lines (18)–(21) show. Finally, in the global pheromone update phase, the best and worst ants will be found and the positive and negative pheromone values on the paths they passed through will be updated, as lines (23)–(28) depict. Once the termination criterion is met, MPTACO will output the best result (line (30)) and terminate the search process.

## Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

## Acknowledgments

The authors would like to thank the editors and anonymous reviewers for their valuable comments and suggestions on the paper that greatly improve the quality of the paper. This work was supported in part by the Ministry of Science and Technology of Taiwan, under Contracts MOST103-2221-E-110-061, MOST103-2221-E-197-034, and MOST103-2221-E-006-145-MY3.

## References

[1] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, Englewood Cliffs, NJ, USA, 1988.

[2] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[3] R. Xu and D. Wunsch II, "Survey of clustering algorithms," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 645–678, 2005.

[4] M. P. Brown, W. N. Grundy, D. Lin et al., "Knowledge-based analysis of microarray gene expression data by using support vector machines," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 97, no. 1, pp. 262–267, 2000.

[5] G. Getz, H. Gal, I. Kela, D. A. Notterman, and E. Domany, "Coupled two-way clustering analysis of breast cancer and colon cancer gene expression data," *Bioinformatics*, vol. 19, no. 9, pp. 1079–1089, 2003.

[6] W. Xu, X. Liu, and Y. Gong, "Document clustering based on non-negative matrix factorization," *Proceedings of the International ACM SIGIR Conference*, pp. 267–273, 2003.

[7] A. Leuski, "Evaluating document clustering for interactive information retrieval," in *Proceedings of the 10th International Conference on Information and Knowledge Management (CIKM '01)*, pp. 33–40, ACM, November 2001.

[8] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: a literature survey," *ACM Computing Surveys*, vol. 35, no. 4, pp. 399–458, 2003.

[9] T. Fromherz, P. Stucki, and M. Bichsel, "A survey of face recognition," MML Technical Report, 1997.

[10] F. Giannotti, M. Nanni, D. Pedreschi, and F. Samaritani, "Webcat: automatic categorization of web search results," in *Proceedings of the 11th Italian Symposium on Advanced Database Systems (SEBD '03)*, pp. 507–518, Cetraro, Italy, June 2003.

[11] P. Ferragina and A. Gulli, "A personalized search engine based on web-snippet hierarchical clustering," in *Proceedings of the 14th International World Wide Web Conference (WWW '05)*, pp. 801–810, May 2005.

[12] H. J. Zeng, Q. C. He, Z. Chen, W. Y. Ma, and J. Ma, "Learning to cluster web search results," in *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 210–217, July 2004.

[13] A. K. Jain, "Data clustering: 50 years beyond $k$-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[14] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the 5th Symposium on Math, Statistics, and Probability*, pp. 281–297, 1967.

[15] M. K. Ng and J. C. Wong, "Clustering categorical data sets using tabu search techniques," *Pattern Recognition*, vol. 35, no. 12, pp. 2783–2790, 2002.

[16] K. Krishna and M. N. Murty, "Genetic $k$-means algorithm," *IEEE Transactions on Systems, Man, and Cybernetics. Part B: Cybernetics*, vol. 29, no. 3, pp. 433–439, 1999.

[17] S. Bandyopadhyay and U. Maulik, "An evolutionary technique based on $k$-means algorithm for optimal clustering in $\mathbb{R}^N$," *Information Sciences*, vol. 146, no. 1–4, pp. 221–237, 2002.

[18] D. W. van der Merwe and A. P. Engelbrecht, "Data clustering using particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation (CEC '03)*, vol. 1, pp. 215–220, December 2003.

[19] X. Cui, T. E. Potok, and P. Palathingal, "Document clustering using particle swarm optimization," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS '05)*, pp. 185–191, IEEE, June 2005.

[20] A. Ghosh, A. Halder, M. Kothari, and S. Ghosh, "Aggregation pheromone density based data clustering," *Information Sciences*, vol. 178, no. 13, pp. 2816–2831, 2008.

[21] C. W. Tsai, K. C. Hu, M. C. Chiang, and C. S. Yang, "Ant colony optimization with dual pheromone tables for clustering," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, pp. 2916–2921, June 2011.

[22] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[23] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 26, no. 1, pp. 29–41, 1996.

[24] P. S. Shelokar, V. K. Jayaraman, and B. D. Kulkarni, "An ant colony approach for clustering," *Analytica Chimica Acta*, vol. 509, no. 2, pp. 187–195, 2004.

[25] C. F. Tsai, C. W. Tsai, H. C. Wu, and T. Yang, "ACODF: a novel data clustering approach for data mining in large databases," *Journal of Systems and Software*, vol. 73, no. 1, pp. 133–145, 2004.

[26] R. J. Kuo, H. S. Wang, T.-L. Hu, and S. H. Chou, "Application of ant $k$-means on clustering analysis," *Computers and Mathematics with Applications*, vol. 50, no. 10–12, pp. 1709–1724, 2005.

[27] R. Tiwari, M. Husain, S. Gupta, and A. Srivastava, "Improving ant colony optimization algorithm for data clustering," in *Proceedings of the International Conference and Workshop on Emerging Trends in Technology (ICWET '10)*, pp. 529–534, February 2010.

[28] E. Akarsu and A. Karahoca, "Simultaneous feature selection and ant colony clustering," *Procedia Computer Science*, vol. 3, pp. 1432–1438, 2011.

[29] L. Jiang, L. Ding, Y. Peng, and C. Zhao, "An efficient clustering approach using ant colony algorithm in mutidimensional search space," in *Proceedings of the International Conference on Fuzzy Systems and Knowledge Discovery*, vol. 2, pp. 1085–1089, July 2011.

[30] X. Liu and H. Fu, "An effective clustering algorithm with ant colony," *Journal of Computers*, vol. 5, no. 4, pp. 598–605, 2010.

[31] T. Niknam and B. Amiri, "An efficient hybrid approach based on PSO, ACO and $k$-means for cluster analysis," *Applied Soft Computing*, vol. 10, no. 1, pp. 183–197, 2010.

[32] M. C. Chiang, C. W. Tsai, and C. S. Yang, "A time-efficient pattern reduction algorithm for $k$-means clustering," *Information Sciences*, vol. 181, no. 4, pp. 716–731, 2011.

[33] S. Kirkpatrick, J. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[34] F. Glover, "Tabu search—part I," *ORSA Journal on Computing*, vol. 1, no. 3, pp. 190–206, 1989.

[35] J. H. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, Boston, Mass, USA, 1992.

[36] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, IEEE, Perth, Australia, November-December 1995.

[37] UCI Datasets, http://archive.ics.uci.edu/ml/datasets/Iris.