K-Ants Clustering- A New Strategy Based on Ant Clustering

Zahra Sadeghi [1], Mohammad Teshnehlab [2], Mir Mohsen Pedram [3]
[1] *Computer Engineering Department, Science and Research Branch, Islamic Azad University - Member of Young Researchers Club*
[2] *Faculty member, Electronic Engineering Department, K.N.Toosi University, Tehran, Iran*
[3] *Faculty member, Computer Engineering Department, Tarbiat-Moallem University, Tehran, Iran*

[1]*z.sadeghi@seiau.ir*, [2] *teshnehlab@eetd.kntu.ac.ir*, [3]*pedram@tmu.ac.ir*

Abstract*-In this article a new method of clustering with artificial ants is proposed. Unlike conventional ant clustering algorithms the true number of clusters must be provided for this algorithm in advance. Clustering is done using groups of ants which are as many as the number of clusters. The goal of each group is to collect members of one cluster. Two new functions are defined inspired from existing pick up and drop down probability functions which are used for inserting and removing loads of ants. Experimental results of the method demonstrated better accuracy in comparison to k-means and ant based clustering algorithm.*

Keywords*:* ant based clustering, ant colony optimization.

1. Introduction

Clustering is an important technique that has been studied in various fields with many applications such as image processing, marketing, data mining and information retrieval. The goal of clustering is partitioning data into some groups so as all the members of one group have a close relationship to each other. This relationship is often expressed as similarity/dissimilarity measurement and is calculated through distance function. Recently, algorithms inspired by nature are used for clustering. The swarm intelligence clustering models and algorithms have advantages in many aspects, such as self-organization, flexibility, robustness, no need of priori information, and decentralization [1]. Ant colony optimization (ACO) is a relatively new and expanding branch of intelligent systems. A feature of these algorithms is that the clustering objective is implicitly defined. Implicit here means that neither the overall objective of the process (i.e., clustering) nor the types of clusters sought are defined explicitly at any point during the clustering process [2]. Some previous works done on ant based clustering include adaptive setting of some parameters[3-6], use of pheromone traces for directing ants to direct ant movement towards promising grid positions[6-7] and the replacement of picking and dropping probabilities by fuzzy rules[8-9],[2].

In this paper, we introduce a new approach for clustering with ants in which the number of clusters must be provided for it in advance. The clustering is done using a square grid. Each ant has a load list that must be filled with the members of one cluster. So every ant is supposed to search for one distinct cluster. Ants use inserting and removing functions for adding a new member to their load list and to remove the wrong members from it. These functions are the modified form of the popular picking up and dropping down functions and we will introduce them in section 3.

2. The General Principle of Ant Clustering

Ant based clustering is a distributed process. The pioneers of this work are Deneubourg *et al*. [10]. Their model is known as basic model (BM) in which ants are simulated by simple agents that randomly move in an environment which is a square grid. Initially, each data object that represents a multi-dimensional pattern is randomly distributed over the 2-D space. Data items that are scattered within this environment can be picked up, transported and dropped by the agents in a probabilistic way. The picking and dropping operations are influenced by the similarity and density of the data items within the ant's local neighborhood. Deneubourg *et al*. proposed a computational model for spatial sorting. Their model is based on the following probability of picking and dropping functions:

$$Pp = \left(\frac{Kp}{Kp + f}\right)^2. \quad (1)$$

$$Pd = (\frac{f}{Kd + f})^2. \quad (2)$$

Where Pp is the probability of picking, Pd is the probability of dropping, and Kp and Kd are constants. f is the perceived fraction of items in the neighborhood of the ant. The probability of picking (Pp) is increased if a data object is surrounded by dissimilar data, or when there's no data in its neighborhood. Also, ants trend to drop data in the vicinity of similar ones, so the probability of dropping (Pd) is increased if ants are surrounded with similar data.

Deneubourg *et al.*'s model was later extended by Lumer *et al.* [11] to allow its application to exploratory data analysis. They modified the probability functions as the followings:

$$Pp(i) = (\frac{Kp}{Kp + f(i)})^2. \quad (3)$$

$$Pd(i) = \begin{cases} 2f(i) & if \quad f(i) < Kd \\ 1 & if \quad f(i) \geq Kd. \end{cases} \quad (4)$$

Where Pp and Pd are as before, i.e., the probability of dropping and picking. In addition, they introduced the density function as the following:

$$f(i) = \max(0.0, \frac{1}{\sigma^2} \sum_{j \in L} (1 - \frac{\delta(i, j)}{\alpha})). \quad (5)$$

Where $\delta(i, j) \in [0,1]$ is the dissimilarity function between two data object i and j. $\alpha \in [0,1]$ is the scaling parameter that permits further adjustments of resulting values. σ^2 is the size of local neighborhood L around the ant's current position. The basic ant clustering algorithm is like the one summarized as bellow [2], [12]:

Let $R \in [0,1]$ be a random number drawn for each use, the basic algorithm is then given by:

Randomly scatter data items on a square grid.

For a number of steps or until a criterion is met, repeat for each ant:

- If the ant is unladen and placed on location l occupied by object o, the object is picked up if $R \leq p_{pick\_up}(o)$.

- If the ant is carrying object o and placed on an empty location l, the object is dropped if $R \leq p_{drop\_down}(o)$.

- Move the ant to a new randomly selected (neighboring) location.

3. The General Idea of K-ants Clustering

Like the basic model our method uses grid for clustering but in contrast to that it doesn't make clusters on the grid. One major problem with the basic model of ant based clustering is that if there were no clear boundary between data objects on the grid then retrieving clusters can be done with ambiguity. Our proposed method uses k ants for finding k clusters and they collect data objects in their load list, so there is no need to the retrieving process. We have considered a grid with s*s cells. The number of cells must be greater than the number of all objects. In the beginning of clustering, we generate ants as many as the number of the clusters. Each ant must find the members of one cluster. All ants move with different speeds on the grid. We have considered a maximum speed for all ants which is set as max_speed=s/3. The step of movement of each ant is generated randomly between 1 and max_speed before each movement. We decrease the max_speed by one every 200 iterations. The density function (6) which was proposed by Handle *et al*. [13], has been used in this study.

$$f(i) = \begin{cases} \max(0.0, \frac{1}{\sigma^2} \sum_{j \in L} (1 - \frac{\delta(i, j)}{\alpha})) & if \quad (1 - \frac{\delta(i, j)}{\alpha}) > 0 \\ 0 & otherwise. \end{cases} \quad (6)$$

Where $\delta(i, j) \in [0,1]$ is the dissimilarity function between two data object i and j. $\alpha \in [0,1]$ is a scaling parameter that permits further adjustments of resulting values. L is the load list of the ant that wants to insert a new item or remove an existing one from it. σ is the size of L. We have also used the probability of picking up and dropping down functions proposed by Lumer & Faieta in the modified form of inserting and removing which are defined according to formula (7) and (8), respectively. The value of the parameters that we have chosen for our experiment is presented in Table 1. These settings are selected because we got satisfying results using them.

$$P_{insert}(i) = \begin{cases} (\frac{f(i)}{K_{ins} + f(i)})^2 & if \quad f(i) < K_{ins} \\ 1 & if \quad f(i) \geq K_{ins} \end{cases} \quad (7)$$

$$P_{remove}(i) = \begin{cases} 0 & if \quad f(i) \geq K_{rem} \\ (\frac{K_{rem}}{K_{rem} + f(i)})^2 & if \quad f(i) < K_{rem} \end{cases} \quad (8)$$

Where K_{ins} and K_{rem} are the inserting and removing constants respectively.

Whenever an ant is located in an occupied cell, it tries to pick up the item and insert it into its load list according to inserting formula. Similarly, if an ant is located in an empty cell, it searches its load list and tries to remove one of its loads according to the removing formula. For the sake of accuracy, in every iteration each ant tries to merge its loads with other ants' load list. For this, each ant checks all of its loads for inserting into other ants' load list using the inserting probability function. In this way the wrongly clustered data can be returned back into their correct clusters and the error of clustering decreases. This is a very important process which prevents ants from falling into local minimum.

4. The Basic K-ants Clustering Algorithm

The detail of our proposed k-ants clustering algorithm is explained in Algorithm 1.

Algorithm 1. Basic form of proposed K-ants algorithm

// Initialization part

1. Consider the number of clusters and generate equal ants
2. Spread the data and ants randomly on the grid and give all ants a random load.

// main part

3. **While** iter < max_iter
4. **For** i=1 to #ants
5. a = Choose a random ant
6. step = Generate a random speed from interval (1,max_speed)
7. new_location=Move (a, step)
8. **If** there is a load l in new_location then
9. Calculate the density function of l according to formula (6). L is the load list of ant a
10. inserting_p = Calculate the inserting probability of load l according to formula (7)
11. r = generate a random number from interval (0, 1).
12. **If** inserting_p > r then
13. Enter the load l into the load list of ant a.
14. **End If**
15. **Else If** there is no load in new_loacation then
16. **For** all loads l of ant a
17. Calculate the density function of l according to formula (6). L is the load list of ant a
18. removing_p = calculate the removing probability of load l according to formula (8)
19. r = Generate a random number from interval (0,1)
20. **If** removing_p>r then
21. remove load l from the load list of ant a
22. **End If**
23. **End For**
24. **End If**

// update the load list of ant a

25. **For** k=1 to #ants
26. **For** l=1 to #load_list(a)
27. **If** k~=a
28. f = Calculate the density function of l according to formula (6). L is the load list of ant k
29. inserting_p = calculate the inserting probability of l according to formula (7)
30. r = Generate a random number from the interval (0,1)
31. **If** inserting_p>r then
32. insert the load l into load list of ant k and remove it from load list of ant a
33. **End**
34. **End If**

35. **End For**

36. **End For**

37. **End For**

38. **End while**

5. Algorithm Modifications

In order to improve the convergence speed and accuracy we have decided some modifications which are discussed in the following subsections.

5.1 Helper Ants

The grid space can be explored very slowly if the number of clusters is much less than the size of the dataset. To avoid this to happen, more than one ant can be used for gathering the members of one cluster. We have called them helper ants. By using helper ants clustering can be done much more quickly because of the cooperation of a number of ants for finding the members of one cluster. In this way every ant must always be aware of all the members of other ants of its group. Each time an ant wants to insert or remove an item it must take other members of its group into account. So the parameter L in formula (6) will change to the members of all load lists of the ants of one group i.e. all helper ants of one group. Note that an ant can only update its load list with an ant of opposite group.

We have used 2 helper ants for each group in our implementation.

Table 1. Parameters and their corresponding values

Parameter	Value
K_{ins}	0.15
K_{rem}	0.15
α	0.5

5.2 Blocking the Empty Cells

After a few iterations, most cells become empty. Stepping on the empty cells can only cause ants to put their loads on them. These loads must then be taken by other ants. This process is too time consuming and prolongs the gathering of members of clusters. To prevent this to happen, after some constant number of iterations, the empty cells become block. No ants can enter a block cell.

5.3 Stopping the Action of Putting on the Grid

Instead of putting down the loads iteratively and then picking them up, we have forbidden the action of putting on the grid after some constant number of iterations. In this way the correction of clusters can only be done by updating the load lists of ants. By this heuristic, the transfer of one load from one ant's load list to another can be done directly and the redundant action of putting on the grid will be avoided.

6. Comparison and Experimental Results

6.1 Evaluation Functions

We have used the F-Measure [14] and Rand Index [15] for evaluating the clusters generated by our algorithm. These functions are defined as the followings [16], [17]:

The F-Measure uses the ideas of precision and recall from information retrieval. Each class i (as given by the class labels of the used benchmark data set) is regarded as the set of n_i items desired for a query; each cluster j (generated by the algorithm) is regarded as the set of n_j items retrieved for a query; n_{ij} gives the number of elements of class i within cluster j. For each class i and cluster j, precision and recall are then defined as (9) and (10) and the corresponding value under the F-Measure is defined as (11).

$$p_{ij} = \frac{n_{ij}}{n_j} \tag{9}$$

$$r_{ij} = \frac{n_{ij}}{n_i}. \tag{10}$$

$$f_{ij} = \frac{(b^2+1).p_{ij}.r_{ij}}{b^2.p_{ij} + r_{ij}}, \tag{11}$$

Where we chose b=1, to obtain equal weighting for f_{ij} and r_{ij}. The overall F-value for the partitioning is computed as (12).

$$F = \sum_i \frac{n_i}{n} \max_j \{ f_{ij} \}, \tag{12}$$

Where n is the total size of the data set. F is limited to the interval [0, 1] and should be maximized.

The Rand Index determines the degree of similarity (in terms of pair wise co-assignments) between the known correct classification U and the solution V

generated by a clustering algorithm. It is defined as (13).

$$R = \frac{a+d}{a+b+c+d}. \quad (13)$$

Where a, b, c and d are computed for all possible pairs of data points i and j and their respective clusters assignment, $C_U(i)$, $C_U(j)$, $C_V(i)$ and $C_V(i)$ according to (14).

$$\begin{aligned} a &= \left|\{i, j \mid C_U(i) = C_U(j) \wedge C_V(i) = C_V(j)\}\right| \quad (14)\\ b &= \left|\{i, j \mid C_U(i) = C_U(j) \wedge C_V(i) \neq C_V(j)\}\right| \\ c &= \left|\{i, j \mid C_U(i) \neq C_U(j) \wedge C_V(i) = C_V(j)\}\right| \\ d &= \left|\{i, j \mid C_U(i) \neq C_U(j) \wedge C_V(i) \neq C_V(j)\}\right| \end{aligned}$$

R is limited to the interval [0, 1] and is to be maximized.

6.2 Comparative Results and Comparison

The datasets we used are summarized in Table 2 and the results of applying the evaluation functions (F-Measure and Rand Index) on them are shown in Table III. It shows means and standard deviations (in parentheses) which is obtained over 30 independent runs with different initializations for the two measures. Additionally, Table 3 shows the error of clustering with respect to the number of wrongly clustered data. In all cases we have compared our results with k-means and ant-based clustering algorithm using LF model.

The results show that unlike k-means algorithm, k-ants algorithm doesn't fall into local minimum and its results are independent to the initialization values. In addition, the clusters obtained from k-ants algorithm are significantly better than k-means and ant clustering algorithm in term of accuracy.

Table 2. summary of the used data sets. dim is the dimensionality, k gives the number of clusters, and size shows the number of members of a cluster.

Name-k-size	Dim	Source
Square5-4*100	2	N([0,0],[2,2]),([0,5],[2,2]),([5,0],[2,2]),([5,5],[2,2])
Iris-3*50	4	UCI

Table 3. Test results on benchmark dataset. Underlined bold face indicates the best and bold face the secondbest result out of the three algorithms.

Iris	K-means	LF	k-ants (proposed method)
Min Error	8	**7**	**5**
Max Error	**19**	**16**	21
Average Error	12.18182	**9.85**	**9.61905**
F-Measure	**0.86665** (0.036583)	0.825911 (0.0148461)	**0.918927** (0.030938)
Rand Index	0.854525 (0.030061)	**0.855422** (0.01814619)	**0.905753** (0.030487)
Total Iteration	**100**	400000	**5000**
Square5	K-means	LF	k-ants (proposed method)
Min Error	**1**	5	**2**
Max Error	24	**10**	**12**
Average Error	8.47	**6.22**	**4.34**
F-Measure	**0.832526** (0.040423)	0.823462 (0.0411472)	**0.862134** (0.022634)
Rand Index	**0.844524** (0.042345)	0.835422 (0.02814619)	**0.875753** (0.020487)
Total Iteration	**100**	400000	**5000**

7. Conclusion

In this paper, we presented a new strategy for clustering using artificial ants in which groups of ants tries to do clustering by inserting and removing operations. The number of groups of ants is equal to the number of clusters which must be specified in prior to clustering. All data objects and ants are spread randomly on the grid. Each ant contains a load list which is initialized with a random object at first. Ants search the grid and try to collect similar data to their load. The load list of ants of each group constructs each cluster and so in contrast to the basic ant clustering algorithms there is no need for the extra process of cluster retrieval from grid. The experimental results dedicated better performance in comparison to k-means algorithm and the LF model of ant clustering. It also outperforms k-means algorithm because it doesn't fall in local minima and its results are not dependant to the initialization part of algorithm.

References

[1] W. bin, S. Zhongzhi, "A clustering algorithm based on swarm intelligence", Proc. of the Int. Conf. on Info-tech and Info-net, Beijing, China, pp. 58-66, 2001.

[2] J. Handl, B. Meyer, "Ant-based and swarm-based clustering, Swarm Intelligence", pp. 95-113, doi: 10.1007/s11721-007-0008-7, 2007.

[3] J. Handl, B. Meyer, Improved ant-based clustering and sorting in a document retrieval interface. In J. J. Merelo, P. Adamidis, & H.-G. Beyer (Eds.), Lecture notes in computer science: Vol. 2439, 2002.

[4] J. Handl, J. Knowles, M.Dorigo, "Ant-based clustering and topographic mapping", Artificial Life, 12(1), 35–61, 2006.

[5] A. L. Vizine, L. N. de Castro, R. R. Gudwin, "Text document classification using swarm intelligence", In International conference on the integration of knowledge intensive multi-agent systems", Piscataway: IEEE Press. problem solving from nature—PPSN VII, pp. 134–139, 2005.

[6] A. L.Vizine, L. N. de Castro, E. R. Hruschka, , R. R. Gudwin, "Towards improving clustering ants: an adaptive ant clustering algorithm" Informatica, 29, 143–154, 2005.

[7] V. Ramos, J. Merelo, "Self-organized stigmergic document maps: environments as a mechanism for context learning", In Proc. of the first Spanish conf. on evolutionary and bio-inspired algorithms, pp. 284–293, 2002.

[8] S. Schockaert, , M. D. Cock, C. Cornelis, E. E. Kerre, "Efficient clustering with fuzzy ants", In D. Ruan, P. D'hondt, M. D. Cock, M. Nachtegael, & E. E. Kerre (Eds.), Applied computational intelligence, proceedings of the 6th international FLINS conference, pp. 195–200, 2004.

[9] S. Schockaert , M. D. Cock, C. Cornelis, , E. E. Kerr , "Fuzzy ant based clustering", In M. Dorigo, M. Birattari, C. Blum, L. M. Gambardella, F. Mondada, & T. Stützle (Eds.), Lecture notes in computer science: Vol. 3172, pp. 342–349, 2004.

[10] J.-L. Deneubourg, S.Goss, N.Franks, C.Detrain, and L.Chretien "The Dynamics of Collective Sorting: Robot-Like Ant and Ant-Like Robot", In J.-A. Meyer & S. Wilson (Eds.), from animals to animats: proc. of the first int. conf. on simulation of adaptive behavior, Cambridge: MIT Press, pp. 356-365, 1991.

[11] E. Lumer, B. Faieta, "Diversity and Adaptation in Populations of Clustering Ants", Cambridge: MIT Press, In D. Cliff, P. Husbands,J.-A. Meyer, & S. W. Wilson (Eds.), From animals to animats: proceedings of the third international conference on simulation of adaptive behavior, pp. 501-508, 1994.

[12] E. Magnus, H. Pedersen, "Ant Colony Clustering & Sorting", 2003.

[13] J. Handl, J. Knowles, M. Dorigo, "Strategies for the increased robustness of ant-based clustering" Self-Organising Applications: Issues, challenges and trends. Springer-Verlag. Lecture Notes in Computer Science, Vol. 2977. pp. 90-104, 2003.

[14] C. V. Rijsbergen, "Information Retrieval, 2nd ed. London", UK: Butterworths, 1979.

[15] W. Rand, "Objective criteria for the evaluation of clustering methods", Journal of American Statistical Association, pp. 846-850. 1971.

[16] J. Handl, "Ant-based methods for tasks of clustering and topographic mapping: extensions, analysis and comparison with alternative methods", Master's Thesis. Chair of Artificial Intelligence, University of Erlangen-Nuremberg, Germany, 2003.

[17] J. Handl, J. Knowles, M. Dorigo, "On the performance of ant-based clustering", Design and Application of Hybrid Intelligent Systems. Amsterdam, The Netherlands: IOS Press. Frontiers in Artificial Intelligence and Applications, vol. 104. pp. 204-21, 2003.