A HYBRID ANT-BASED CLUSTERING ALGORITHM

Marianne Chong and Mylini Munusamy
Monash University Malaysia, No. 2, Jln Kolej, Bandar Sunway, 46150 Petaling Jaya, Selangor, Malaysia

Abstract: This research examines the ant-based clustering method as an alternative to k-means algorithm. Of particular interest is the AntClust which is modeled after the nestmate recognition system of real ants. We propose an algorithm called the Hybrid Ant-based Clustering Algorithm (HACA) which is a hybrid approach for unsupervised clustering problems. This algorithm combines the features of AntClust and k-means. HACA employs three different strategies – the blacklist strategy, the sniffing strategy and the nests fusion method, to improve the performance of the algorithm. We have conducted experimental investigations to demonstrate the effectiveness of HACA and the results have shown the advantages of the three strategies proposed as well as the improved performance of HACA compared to two other algorithms.

Key words: ant-based clustering, k-means algorithm,

1. INTRODUCTION

Clustering is an unsupervised task that seeks to identify natural groupings in data. The k-means algorithm is a traditional clustering algorithm that is commonly used where the advantages are its simplicity and speed. However, the shortcomings of k-means are that there is a need to supply an initial partition of the data set, the tendency to converge at a local optimum if the initial partition provided is not good and the assumption that all the clusters are of spherical shape. As a solution to these problems, various algorithms based on the behavior of real ants have been proposed. Ant-based algorithms have the advantages of not requiring an initial partition of the data set, being able to identify clusters of arbitrary shapes because of the bottom-up approaches of the algorithms as well as having a stochastic nature that allows the exploration of the data set. A hybrid algorithm that combines the clustering approach of k-means algorithm and an ant-based algorithm would counterbalance the advantages and disadvantage of both approaches.

In this research we propose a hybrid ant-based clustering algorithm (HACA) for unsupervised clustering problems. This algorithm combines the ant-based clustering algorithm with the k-means algorithm. As a measure of the performance of the algorithm, we have conducted experiments using differing data sets. Additionally, we also compared HACA against AntClust [8] and k-means.

The rest of the paper is organized as follows. The next section presents a brief literature review of various ant-based clustering algorithms. Section 3 discusses the AntClust [8] algorithm followed by a description of HACA and the three strategies that we have proposed to improve the algorithm. Section 4 details the experimental investigations that we have conducted to demonstrate the effectiveness and performance of HACA. Finally, Section 5 concludes this paper.

2. RELATED WORK

Various works on ant-based clustering algorithms that have been proposed based on either corpse clustering and larval sorting behaviors or the self-assembly behavior of real ants. The use of artificial ants for data clustering was first pioneered by Deneubourg et al [1] where they proposed a model based on the behaviors of corpse clustering and brood sorting in real ant colonies. This model is based on the idea of picking and dropping isolated objects at location where more objects of the same type are present. Lumer and Faieta [10] generalized Deneubourg's model for application on exploratory data analysis where they introduced a function to accommodate the clustering and sorting of complex objects. There are also algorithms that make use of artificial pheromone for ant clustering and sorting such as TermitAnt [14].

Additionally, there are hybrid algorithms that combine the Lumer and Faieta [13] algorithm with traditional clustering algorithms such as k-means and fuzzy c-means. The AntClass [11] is a hybrid algorithm that combines the Lumer and Faieta [10] algorithm with the k-means algorithm. This algorithm provides the initial partition for the k-means algorithm whilst the k-means algorithm refines the clustering results from the ant-based clustering algorithm. Schockaert et al. [13] proposed a simplified version of AntClass combining fuzzy IF-THEN rules that allows an ant to pick up either an object or a heap. The main advantage of ant-based algorithms is that it is able to generate a reasonably good clustering of the data set without any prior information about the data set.

3. HACA

We propose a hybrid approach called HACA that combines the features of the AntClust algorithm [8] with the k-means algorithm to counterbalance the advantages and disadvantages of both approaches.

3.1 AntClust

AntClust is an ant-based clustering algorithm that is modeled after the nestmate recognition system where the ants are used to represent items to be clustered (see Figure 1). The nestmate recognition system is described as a phenomenon that is "typically manifested by rejecting alien intruders" [8].

During the meeting phase, two ants are randomly picked at each iteration and a meeting is simulated between them. The outcome of the meeting is dictated by one of six behavioral rules. After the meeting phase, the nests formed are filtered in the nests deletion phase and only the meaningful nests are retained. Each artificial ant is coded as a set of parameters consisting of a $Label_i$, $Genome_i$, $Template_i$, M_i, M_i^+, and A_i.

The label $Label_i$, denotes the belonging nest and is initialised to 0 at the start of the algorithm. The label changes throughout the meeting phase until the ant finds the nest that best matches its genome. The genome $Genome_i$, corresponds to an item in the input data set and enables an ant to generate its own odour. It is used to learn and update its template. During meetings, ants compare their genome to evaluate the similarity between themselves. The template $Template_i$, is the acceptance threshold for an ant. This is one of two thresholds that are used to determine acceptance or rejection when an ant meets with another ant – the similarity between the genomes of the two ants has to be higher than $Template_i$ and $Template_j$ in order for acceptance to occur. $Template_i$ is learned during the initialisation period, in which an ant meets with a specified number of other ants, and each time evaluates the similarity between their genomes. After $Template_i$ has been learned, it is updated following every meeting realised by the ant regardless of the outcome, as the similarities observed may have changed.

```
(1)  Initialisation of the ants:
(2)  $\forall$ ants $i \in [1, N]$
(3)      $Genome_i \leftarrow i^{th}$ object of the data set
(4)      $Label_i \leftarrow 0$
(5)      $Template_i$ is learned during $N_{App}$ iterations
(6)      $M_i \leftarrow 0, M_i^+ \leftarrow 0, A_i \leftarrow 0$
(7)  $Nb_{Iter} \leftarrow 75 * N$
(8)  Simulate $Nb_{Iter}$ meetings between two randomly chosen ants
(9)  Delete the nests that are not interesting with a probability $P_{del}$
(10) Re-assign each ant that has no more nest to the nest of the most similar ant
```

Fig. 1. The AntClust Algorithm

The estimator M_i, indicates the proportion of meetings with nestmates and consequently enables an ant to have an estimation of the size of its nest. It increases each time an ant meets another ant with the same label, and decreases otherwise. The estimator M_i^+, reflects the proportion of positive meetings with nestmates. As positive meetings happen only when there is acceptance between ants, M_i^+ indicates the strength of an ant's integration in its own nest through acceptance from its nestmates. It is similar to M_i but with the additional condition that the nestmates accepted each other. M_i increases when there is acceptance between an ant and an encountered nestmate, and decreases otherwise. The age A_i, keeps track of the number of meetings that an ant has had. It is used to update the values of the maximal and mean similarities, and thus the value of *Template$_i$*. A_i increases each time an ant meets another ant. These parameters evolve according to a set of behavioural rules, as shown in Figure 2.

Once the ants have learned their templates, two ants at a time are picked to meet. These meetings are used to generate the nests that will define the final clusters. The first step in a meeting is the evaluation of similarity between the pair of ants. Then, each ant compares the evaluated similarity against its own template (i.e. the acceptance threshold). This comparison is also known as the principle of acceptance and rejection between two ants [12]. The outcome of this comparison (i.e. acceptance or rejection) and the current state of each ant are used to determine the correct rule to apply to the meeting, as shown in Figure 2. Briefly, the actions taken depending on the rules are: Rule 1 results in the creation of a new nest that consists of both the ants; Rule 2 joins the nestless ant with the nest of the other ant; Rule 3 increments the estimators M and M^+ for both ants; Rule 4 ejects the less integrated ant from the nest; Rule 5 absorbs the ant from the smaller nest into the bigger nest; and Rule 6 is the default rule where no action is taken.

The advantages of AntClust are that it does not need any initial knowledge of the data set such as the number or shape of the expected clusters in order to converge; this is unlike many other clustering algorithms such as k-means that require an initial partition of the data set prior to execution. It is able to find a partition that is as close as possible to the natural partition of the data set regardless of artificial or real data sets. It is able to find clusters of various shapes and sizes since AntClust adopts a

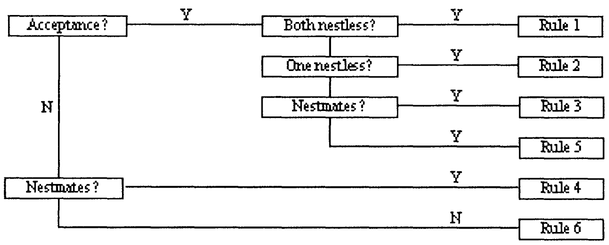

Fig. 2. The process of deciding which behavioral rule to apply

bottom-up approach towards clustering, that is, the clusters are formed from interactions between the ants. It shows favourable scaling behaviour to large and complex data sets. It is easily adapted to any type of data such as numeric or binary provided a suitable similarity measure is defined that takes two items as input and outputs a similarity value between 0 and 1.

3.2 HACA

Although AntClust has its advantages, there is a weakness of this approach. The stochastic nature of AntClust results in an inconsistency of the solutions obtained. Subsequent runs of the algorithm on the data set are likely to produce different clusters than in previous runs. HACA attempts to solve this problem with simple methods that will also improve the performance of the algorithm by producing more accurate classifications. Figure 3 shows the algorithm for HACA where we have introduced a blacklist strategy, a sniffing strategy and the nests fusion method. Apart from introducing these strategies, we have also made modifications to the *Template* and estimator M parameters. We will describe these strategies in the following sections.

3.2.1 Blacklist Strategy

The objective of this strategy is to reduce the number of meetings that is required for HACA to converge. Lenoir et al [9] suggested that the mechanism of recognition relies on detecting difference rather than similarities, therefore an ant has a more pronounced reaction towards its enemies than towards it own nestmates. In this strategy, every ant has the ability to remember information from past meetings with other ants,

```
(1)  Initialisation of the ants:
(2)  ∀ ants $i \in [1, N]$
(3)     $Genome_i \leftarrow i^{th}$ object of the data set
(4)     $Label_i \leftarrow 0$
(5)     $Template_i$ is learned during $N_{App}$ iterations
(6)     $M_i \leftarrow 0, M_i^+ \leftarrow 0, A_i \leftarrow 0$
(7)     $Blacklist_i \leftarrow$ empty
(8)  $Nb_{Iter} \leftarrow ½ * \alpha * N$
(9)  Simulate $Nb_{Iter}$ meetings between two randomly chosen ants, $i$ and $j$
(10)    Check that $j \notin Blacklist_i$
(11)    If no acceptance then
(12)       Predict whether to blacklist ants
(13) Compute for each nest $j \in [1, Nb_{nest}]$, the probability to be deleted $P_{del}(j)$
(14) Test the probability $P_{del}(j)$ for each nest $j \in [1, Nb_{nest}]$ against $\beta$
(15) Delete the nests whose tests are unsuccessful
(16) Perform sniffing on the remaining nests
(17) Re-assign each nestless ant to the nest of the most similar ant to itself
(18) Merge similar nests together
```

Fig. 3. High-level description of HACA.

particularly if it is a negative meeting. These ants are the blacklisted ants. Using these stored identities, an ant can be prevented from making an unnecessary meeting with a blacklisted ant, thus reducing the number of meetings. These meetings are unnecessary because there is little contribution towards the formation of the clustering structure. The incorporation of the blacklist strategy into HACA is likened to keeping a distributed memory of exploration that not only optimises the algorithm in terms of efficiency, but also facilitates the algorithm's response to changing conditions because it can now make use of previous knowledge learnt.

3.2.2 Sniffing Strategy

The nests that are formed using AntClust may contain ants that are not necessarily similar to each other. In fact, it is even possible that some of the ants in the same nest have nothing in common with one another. Additionally, if high-dimensional data is used, this problem is further aggravated as there are more chances for items to be misclassified. We propose a sniffing strategy as an attempt to overcome this weakness. The aim of the sniffing strategy is to maximise the similarity of ants in a nest by removing misplaced ants that exist in the nest. This is done by enforcing central coordination in the algorithm at the nest level. We take advantage of the principle of acceptance and rejection between ants as well as the ejection of ants from a nest in our strategy. A few ants, called "sniffer ants" that are the most integrated in their nest will be in-charge of the quality control in their nest. This sniffing strategy is inspired by the "elitist ants" introduced by Dorigo et al [3].

3.2.3 Nests Fusion Method

We found that the performance of AntClust over consecutive runs on the same data tends to vary [8]. Therefore, we have proposed the nests fusion method as an attempt to minimise the problem of inconsistent clusters with a better estimation of the number of expected clusters in the data set. It compensates for the inadequacies of Rule 5 and the nest deletion method. This method adopts a centroid-based approach, similar to the k-means algorithm. The difference is that nests fusion deals with collections of items as opposed to k-means which deals with individual items.

3.3 Advantages of HACA

We briefly describe some of the advantages of HACA. HACA combines the strengths of bottom-up clustering of AntClust with the top-down clustering of k-means. Bottom-up clustering is good at identifying small clusters, however it provides sub-optimal performance for identifying large clusters. Conversely, top-down clustering is good at identifying a few large clusters, but poor at smaller clusters. Hence, HACA is able to identify clusters of all sizes. The blacklist strategy enables HACA to run on less iterations per data set compared to AntClust. The artificial ants in HACA have the memory to remember the enemies that they have encountered before so that repeat meetings with enemies are avoided. The sniffing strategy functions to find misplaced ants that are then ejected so that they can be re-assigned at a later stage. With the three strategies proposed, HACA is more deterministic as each strategy introduces some degree of deterministic behaviour into the algorithm that reduces the stochastic nature of ant-based clustering algorithms. In the next section, we will describe the experiments that we have conducted to evaluate the effectiveness of HACA.

4. EVALUATION OF HACA

We briefly describe the setup of experiments conducted. All experiments have been conducted using data sets whose classifications are known. We have conducted experiments of the three proposed strategies individually and as a whole algorithm. The experiments were conducted using artificial and real data sets. The artificial data sets used are the 40D-60C data [2], Banana data [2, 5] and the Square data [6] and the real data sets include Iris [7], Soybean [7], Thyroid [7], Wisconsin [7] and Ruspini [12].

We evaluate the algorithm based on the two main criteria – accuracy and consistency. We measure accuracy using two evaluation functions – F-Measure [6, 15] and C-Success. The C-Success is adapted from the clustering success by Labroche et al [8] where we have simplified the measure with a reversal of the values used. The consistency is measured as a relative measure where we evaluate the consistency of HACA's performance over consecutive runs in comparison to other clustering algorithms on the same data sets. Additionally, we have also conducted experiments to determine the optimal values for the parameters used in HACA.

4.1 Overall Performance

We have conducted six individual tests on the eight test data sets as mentioned above. The information collected from each run includes the F-Measure, the C-Success and the mean clusters formed. Figure 4 shows the clustering quality results for each individual strategy.

All three strategies demonstrate similar results in terms of F-Measure and C-Success, especially on the 40D-6C, Square, Iris and Wisconsin data sets. The nests fusion strategy showed the best results in estimating the number of expected clusters for all data sets. It also showed consistent results at estimating close to the correct partitioning over consecutive runs. These results are expected of the nests fusion strategy as its purpose is to join highly similar clusters together. Overall, we found that the nests fusion method gives the best results, followed by the blacklist strategy and the sniffing strategy.

4.2 Comparison against AntClust and k-means

We also compare HACA against the two clustering algorithms it evolves from: AntClust and k-means. Thus providing us with the opportunity to find out how successful the hybrid approach is compared with its predecessors. Our implementation of k-means is based on the batch version, where cluster centers are only recomputed after the assignment of all the data items. In this experiment, we have set the initial partition to be the correct number of clusters corresponding to each data set. As for AntClust, we have implemented AntClust based on the paper by Labroche et al [8]. We have evaluated the clustering methods over the eight test data sets, where for each data set we perform fifty independent runs for each method and compute the F-Measure, C-Success and the mean number of clusters found. The results are summarized in Figure 5.

From the results, we may conclude that HACA performs well on all the data sets, both artificial and real, despite having the disadvantage of not being provided with prior knowledge of the data set. It also reveals that HACA performs better than AntClust on all data sets. This confirms the effectiveness of the three strategies proposed. The strategy that contributes the most significant difference in performance between AntClust and HACA is the nests fusion method as it enhances the ability of HACA to estimate the number of expected clusters in the data set. In comparing the number of clusters found by HACA as opposed to AntClust, we found that the number of clusters found by HACA for all the data sets is closer to the correct

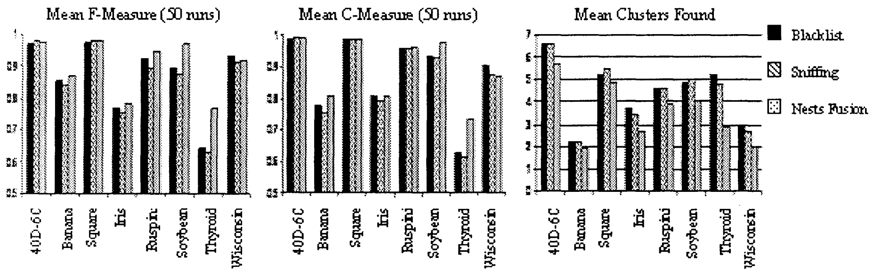

Fig. 4. Mean F-Measure, C-Success and mean clusters for blacklist, sniffing and nests fusion. .

partitioning than that found by AntClust. As for the consistency of the algorithms' performance, HACA's consistency is comparable to k-means where the values of standard deviation for all the data sets are not much larger than those of k-means.

Our experimental evaluations have shown that HACA is capable of generating quality clustering on both artificial and real data sets. In our comparisons, HACA performed better on all data sets, particularly compared to AntClust. The consistency of HACA is generating quality solutions consecutive runs is comparable to k-means for which the number of clusters is provided.

5. CONCLUSION

We have presented our investigation of the use of ant-clustering algorithms for the clustering problems. In particular, we have proposed a new hybrid algorithm that has been developed as a solution for clustering problems. This algorithm, Hybrid Ant-based Clustering Algorithm (HACA), combines the approaches of AntClust [8] and a standard k-means algorithm. In particular, we have introduced three strategies to improve the algorithm – the blacklist strategy, the sniffing strategy and the nests fusion method. This is done to reduce the stochastic nature of the ant-based approach. To evaluate the performance of HACA, we have conducted experimental investigations to determine the effectiveness of the proposed strategies. Additionally, we have also compared the performance of the HACA against AntClust and k-means. The results in both experiments have demonstrated the effectiveness of the three strategies as well as have shown HACA to perform better.

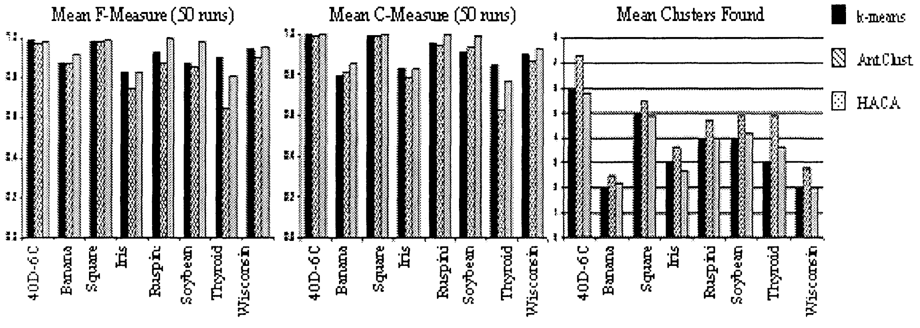

Fig. 5. Mean F-Measure, C-Success and number of clusters for k-means, AntClust and HACA.

REFERENCES

1. Deneubourg, J.-L., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., & Chretian, L. (1991), "The Dynamics of Collective Sorting: Robot-Like Ant and Ant-Like Robot", In: Meyer, J. A., and Wilson, S. W., (eds) *Proceedings First Conference on Simulation of Adaptive Behavior: From Animals to Animals*, MIT Press, Cambridge, MA, pp. 356-365.
2. Dimitriadou, E., Weingessel, A., & Hornik, K. (1999), "A voting-merging clustering algorithm", Working Paper 31, SFB "Adaptive Information Systems and Modeling in Economics and Management Science", Available: http://www.wu-wien.ac.at/am.
3. Dorigo, M., Maniezzo, V., & Colorni, A. (1996), "The Ant System: Optimization by a Colony of Cooperating Agents", *IEEE Transactions on Systems, Man, and Cybernatics-Part B*, vol. 26, no. 1, pp. 1-13.
4. Fred, A. L. N. (2001), "Finding Consistent Clusters in Data Partitions", *Proceedings of the 2nd International Workshop on Multiple Classifier Systems*, July 2-4 2001, LNCS, vol. 2096, Springer-Verlag Heidelberg, London, pp. 309-318
5. Frossyniotis, D., Pertsekalis, M., & Stafylopatis, A. (2002), "A Multi-clustering Fusion Algorithm", In: Vlahavas, I. P. & Spyropoulos, C. D. (eds.) Proceedings of the Second Hellenic Conference on AI: Methods and Applications of Artificial Intelligence, April 11-12, 2002, LNCS, vol. 2308, Springer-Verlag Heidelberg, London, UK, pp. 225-236
6. Handl, J., Knowles, J., & Dorigo, M. (2003), "Ant-based clustering: a comparative study of its relative performance with respect to K-means, average link and 1D-SOM", Technical Report TR/IRIDIA/2003-24. IRIDIA, Universite Libre de Bruxelles, Belgium, Available: http://dbkweb.ch.umist.ac.uk/handl/.
7. Hettich, S. & Bay, S. D. (eds) (2000, April 26 – last update), "The UCI KDD Archive", (*University of California, Irvine*), Available: http://kdd.ics.uci.edu
8. Labroche, N., Monmarché, N., & Venturini, G. (2003), "AntClust: Ant Clustering and Web Usage Mining", *Genetic and Evolutionary Computation Conference*, July 12-16 2003, Lecture Notes in Computer Science, vol. 2723, Springer-Verlag Heidelberg, London, UK, pp. 25-36.
9. Lenoir, A., Fresneau, D., Errard, C., & Hefetz, A. (1999), "Individuality and Colonial Identity in Ants: The Emergence of the Social Representation Concept", In: Detrain, C., Deneubourg, J.-L., and Pasteels, J., (eds) *Information Processing in Social Insects*, Birkhäuser Verlag Basel, Switzerland, pp. 219-237.
10. Lumer, E., & Faieta, B. (1994), "Diversity and Adaptation in Populations of Clustering Ants", *Proceedings Third International Conference on Simulations of Adaptive Behavior: From Animals to Animals 3*, MIT Press, Cambridge, MA, pp. 499-508.
11. Monmarché, N., Slimane, M., & Venturini, G. (1999), "AntClass: discovery of clusters in numeric data by an hybridization of an ant colony with the Kmeans algorithm", *Technical Report 213*, Laboratoire d'Informatique, E3i, University of Tours, France.
12. Ruspini, E. H. (1970), "Numerical methods for fuzzy clustering", *Information Science*, vol. 2, no. 3, pp. 319–350. Results summarised in Kaufman and Rousseeuw (1999).
13. Schockaert, S., De Cock, M., Cornelis, C., & Kerre, E. E (2004), "Efficient Clustering with Fuzzy Ants", In: Ruan, D., D'Hondt, P., De Cock, M., Nachtegael, M., Kerre, E. E., (eds), *Applied Computational Intelligence*, World Scientific Press, pp. 195-200.
14. Sherafat, V., de Castro, L. N., & Hruschka, E. R. (2004), "TermitAnt: An Ant Clustering Algorithm Improved by Ideas from Termite Colonies", In: Pal N. R., Kasabov, N., Mudi, R. K., Pal, S., and Parui, S. K., (eds) *11th International Conference on Neural Information Processing*, November 22-25 2004, LNCS, vol. 3316, Springer-Verlag Heidelberg, London, UK, pp. 1088-1093.
15. Witten, I. H. & Frank, E. (2000), *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*, Morgan Kaufman Publishers, USA.