A Clustering Algorithm Based on the Ants Self-Assembly Behavior

H. Azzag[1], N. Monmarché[1], M. Slimane[1], C. Guinot[2], and G. Venturini[1]

[1] Laboratoire d'Informatique de l'Université de Tours,
École Polytechnique de l'Université de Tours - Département Informatique
64, Avenue Jean Portalis, 37200 Tours, France.
Phone: +33 2 47 36 14 14, Fax: +33 2 47 36 14 22
hanene.azzag@etu.univ-tours.fr
{monmarche,slimane,venturini}@univ-tours.fr
http://www.antsearch.univ-tours.fr/webrtic
[2] C.E.R.I.E.S.,
20 rue Victor Noir, 92521 Neuilly sur Seine Cedex.
christiane.guinot@ceries-lab.com

Abstract. We have presented in this paper an ants based clustering algorithm which is inspired from the self-assembling behavior observed in real ants. These ants progressively become connected to an initial point called the support and then successively to other connected ants. The artificial ants that we have defined similarly build a tree where each ant represents a node/data. Ants use the similarities between the data in order to decide where to connect. We have tested our method on numerical databases (either artificial, real, and from the CE.R.I.E.S.). We show that AntTree improves the clustering process compared to the Kmeans algorithm and to AntClass, a previous approach for data clustering with artificial ants.

1 Introduction

Natural systems have evolved in order to solve many problems that can be related to the data clustering problem. For instance, different species have developed social behaviors to tackle the problem of gathering objects or individuals, like in brood sorting or cemetery organization in real ants [3]. Therefore, several studies involve ants behavior and data clustering [9] [7][4] [11]. We are interested in this paper in showing how to adapt a new biological model to this clustering problem where the data must be hierarchically organized in a tree. This model is based on the ability of ants to build live structures with their bodies [8]. Each ant represents a data and is initially placed on a fixed point, i.e. the root of the tree that we will call the support in the following. The behavior of an ant consists in moving on already connected ants and in connecting itself at a convenient location in the tree. This behavior is directed by the local structure of the tree and by the similarity between data represented by ants. When all ants are connected, the resulting tree can be interpreted as a partitioning of the data in order to solve a clustering problem [6].

W. Banzhaf et al. (Eds.): ECAL 2003, LNAI 2801, pp. 564–571, 2003.

The remainder of this paper is organized as follows: in section 2, we give an overview of the underlying biological model. In section 3, we present the details of the AntTree algorithm. Its properties and comparative results are given in section 4. In section 5, we finally draw some conclusions and describe future evolutions of this method, like its use in Web portal automatic construction.

2 Self-Assembly Behavior in Real Ants

Ants are able to build mechanical structures by a self-assembling behavior. Biologists observe for instance the formation of drops of ants [12], or the building of chains of ants [8]. These types of self-assembly behaviors have been observed with *Linepithema humiles* Argentina ants and African ants of gender *Oecophylla longinoda*. The goal of drop structures built by *L. humiles* is today still obscure. This ability has been recently experimentally demonstrated [12]: ants fix themselves by mean of their tarsus. The drop can sometimes fall down. For *Oecophylla longinoda* ants, it can be observed that two types of chains are built: on the one hand chains of ants fixed with their tarsus are used to cross an empty space, and on the other hand, chains of ants hung by their mandibles and their petioles to build their nest [8]. In both cases, these structures disaggregate after a given time.

Our computer model especially uses the following principles of the ants self-assembly behavior: ants start building the structure from a fixed support (stem, leaf,...) and they may move on the structure in order to become connected. All positions can be reached but for instance in the case of chains, ants preferably fix themselves at the end of the chain because they are attracted by gravity or by the object to reach. Most of the ants which are connected to the structure can not move anymore. In the case of a chain of ants, this corresponds to ants placed in the middle of the chain. A small proportion of connected ants may easily become disconnected from the structure, like for instance the ants placed at the end of a chain. Therefore one may observe a growth but also a decreasing of the structure.

In general, the motivation for using bio-inspired clustering techniques is twofold: they can avoid local minima thanks to their probabilistic behavior and they may produce high quality results without any prior knowledge of data structure (such as the number of clusters or an initial partition). In this work, in addition to these motivations, we are especially interested in showing that this new biological model may be a promising technique for achieving tree-based clustering. One should also notice that ants have been used to build specific kind of trees in [1] but with an ACO algorithm.

3 The Proposed Algorithm: AntTree

3.1 Global Principles

To obtain a partitioning of the data[6], we build a tree where nodes represent data and where edges remain to be discovered. Each data can be described

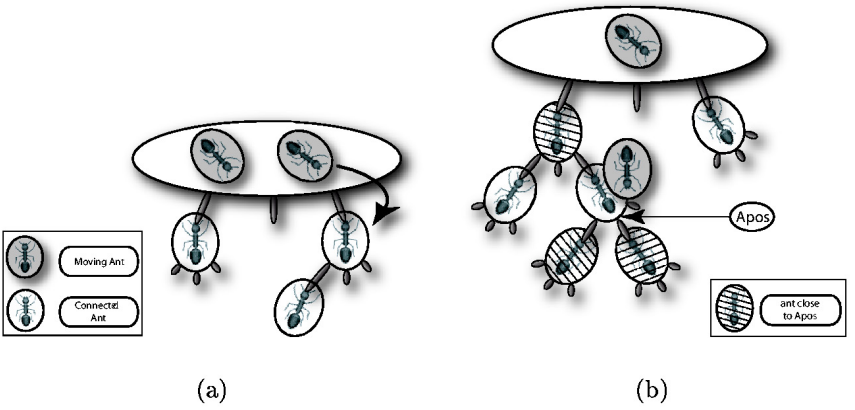


Fig. 1. Representing connected and moving ants (a), and an ant's neighborhood (b)

by any representation language provided that there exists a similarity measure $\mathrm{Sim}(i,j)$ between two data (d_i, d_j), $i \in [1,N]$, $j \in [1,N]$. This function must return values in $[0,1]$ where 0 means that d_i and d_j are totally different and 1 means that they are identical. The main principles of our algorithm called AntTree are the following (see figure 1(a)): the root of the tree (the support) is represented by a node a_0. Ants gradually become connected to this initial node, and then successively to the ants fixed to this node, and so on until all ants are attached to the structure (AntTree stopping criterion). Ants move over the other connected ants and decide where to connect to the structure according to the value returned by $\mathrm{Sim}(i,j)$ and according to the local neighborhood of the moving ants. We consider that each ant $a_i, i \in [1,N]$ has one outgoing link and no more than $L_{\max}$ incoming links (the structure is a tree with maximum degree of $L_{\max}$). We define for each ant a_i a similarity threshold $T_{\mathrm{Sim}}(a_i)$ and a dissimilarity threshold $T_{\mathrm{Dissim}}(a_i)$ that will be locally updated by a_i. These thresholds will be used to determine whether the data d_i represented by a_i is sufficiently similar or sufficiently dissimilar with an other data represented by an other ant.

During the building of the structure, each ant a_i will be either moving on the tree or will be connected to the tree. In the first case, we denote by a_{pos} the ant (or the support) over which a_i is located (and moving). a_i is completely free to move on the support or toward another ant within the neighborhood of a_{pos}. This neighborhood is defined by all ants connected to a_{pos}, considering that edges are undirected (see figure 1(b)). At each step, an ant a_i is selected in a sorted list of ants (we will explain how this list is sorted in section 4.3) and will connect itself or move according to the similarity with its neighborhood.

3.2 Ants Local Behavior

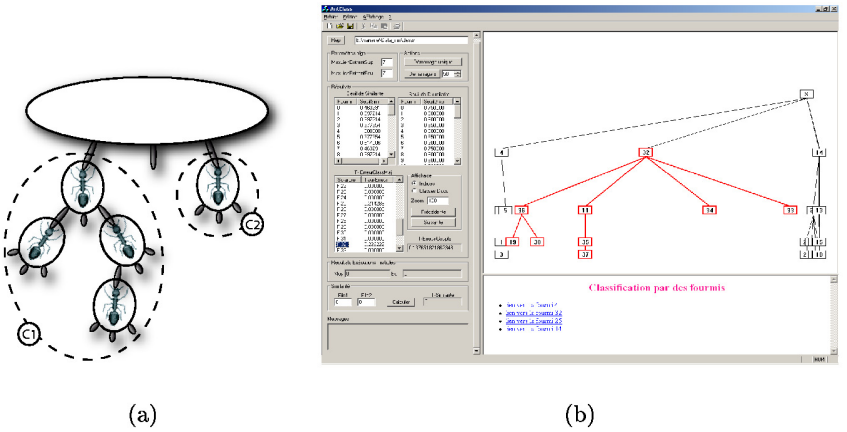
The first ant is straight connected to the support a_0. Next, for each ant a_i, two cases need to be considered. The first case is when a_i is on the support. Let a^+ denotes the ant which is the most similar to a_i among the ants already connected to the support. If a_i is similar enough to a^+ according to its similarity threshold (i.e. $\mathrm{Sim}(a_i, a^+) \geq T_{\mathrm{Sim}}(a_i)$), then a_i is moved toward a^+ in order to be possibly clustered in the same sub-tree, i.e. the same cluster. Else (i.e. a_i is not similar enough to a^+), if a_i is dissimilar enough to a^+ (i.e. $\mathrm{Sim}(a_i, a^+) < T_{\mathrm{Dissim}}(a_i)$), then it is connected to the support. This means that we create a new sub-tree, where ants will be as much dissimilar as possible to the ants in the other sub-trees connected to a_0. If no incoming links are available for a_0, then a_i is moved toward a^+. Finally, if a_i is not similar or dissimilar enough to a^+, we update its thresholds with $T_{\mathrm{Sim}}(a_i) \leftarrow T_{\mathrm{Sim}}(a_i) * 0.9$ and $T_{\mathrm{Dissim}}(a_i) \leftarrow T_{\mathrm{Dissim}}(a_i) + 0.01$. a_i becomes more tolerant and increases its probability to be connected the next time it will be considered (in the meantime, other ants will have changed the tree). We have experimentally chosen the values 0.9 and 0.01 because they provide good results. The similarity threshold must be decreased with a higher rate than the dissimilarity threshold is increased because of the distribution of similarities.

The second case is when a_i is on an other ant denoted by a_{pos} (a^+ also denotes the ant which is the most similar to a_i among the ants connected to a_{pos}). If there is a free incoming link for a_{pos} and if a_i is similar enough to a_{pos} (i.e. $\mathrm{Sim}(a_i, a_{pos}) \geq T_{\mathrm{Sim}}(a_i)$) and dissimilar enough to ants connected to a_{pos} (i.e. $\mathrm{Sim}(a_i, a^+) < T_{\mathrm{Dissim}}(a_i)$), then a_i is connected to a_{pos}. In this case, a_i represents the root of a new sub-tree/sub-cluster below a_{pos}. Its dissimilarity with other ants directly connected to a_{pos} is such that sub-clusters of a_{pos} will be well "separated" from each others (while being similar to a_{pos}). Else, a_i is randomly moved toward a neighbor node of a_{pos} and its thresholds are updated in the same way as in the previous case. So, a_i will move in the tree to find a better location where to be connected. The algorithm ends when all ants are connected.

4 Results

4.1 Testing Methodology

We have used databases in which data are represented with numerical attributes and one class label. Some databases are artificial and have been generated with gaussian and uniform laws (Art1,..., Art8). Other databases (Iris, Wine, Glass, Pima, Soybean and Thyroid) are taken from the the Machine Learning Repository [2] and correspond to standard benchmarks. Finally, we have used the data from the CE.R.I.E.S. that concern the healthy human skin domain [5]. For all these data, there exits a class label which is not given to the clustering methods but which we use for evaluation purposes. This is done by comparing the real partitioning of the data with the obtained one. We use a clustering error measure Ec which evaluates the proportion of misclassified couples of data: for all couples



(a) (b)

Fig. 2. Interpreting the tree as "flat" clusters (a) and tree visual and interactive exploration using for instance hyperbolic zooming (b)

(d_i, d_j), increase the error Ec if d_i and d_j have been clustered in the same class (while being in different real classes) or if the two data have been separated (while they belong to the same real class). We use an Euclidean distance as a similarity measure (data are previously normalized between 0 and 1).

4.2 Interpreting the Results

Results can be interpreted in different and complementary ways. First, the obtained tree-structured partitioning can be interpreted as a "flat partitioning", with the aim for instance of comparing our approach with other "flat clustering" methods (like the Kmeans in the next section). In this case (see figure 2(a)), sub-trees directly connected to the support a_0 are interpreted as clusters. But the tree-structured organization of the data can be useful for the interpretation of the results, which is one of the advantages of hierarchical clustering over flat clustering. As shown in figure 2(b), the interface allows the user to interactively visualize the hierarchy of data using an hyperbolic display. The user may click on data and dynamically change the display. For instance, as one goes deeper into the tree, the classification error decreases as expected. The user may detect sub-structures in the data, like for instance a class which is divided into several sub-classes. This would not be possible with flat clustering.

First, we have tried to find the best strategy for data initial sorting. The initialization step of AntTree influences the results, in particular because the first ants (i.e. first data in the database) will be in general connected first to the support. Since the data that we have used are generally sorted by their class label in the databases, this may completely bias the results. So we have sorted the data in random order, but also with more clever ordering methods: for each data, one may measure its mean similarity with other data. This mean

Table 1. Results obtained by AntTree for different initial order of data, averaged over 50 runs. Ec denotes the classification error and K' the number of classes. σ_{Ec} and $\sigma_{K'}$ are the corresponding standard deviations.

Database	AntTree (decreasing order)		AntTree (increasing order)		AntTree (random order)			
	Ec $[\sigma_{Ec}]$	K' $[\sigma_{K'}]$	Ec $[\sigma_{Ec}]$	K' $[\sigma_{K'}]$	Ec $[\sigma_{Ec}]$	K' $[\sigma_{K'}]$	K	N
Art1	0.75 [0.00]	1 [0.00]	0.17 [0.00]	4 [0.00]	0.44 [0.01]	2.36 [0.08]	4	400
Art2	0.50 [0.00]	1 [0.00]	0.18 [0.00]	3 [0.00]	0.27 [0.01]	1.94 [0.03]	2	1000
Art3	0.58 [0.00]	1 [0.00]	0.21 [0.00]	3 [0.00]	0.37 [0.01]	2.26 [0.08]	4	1100
Art4	0.43 [0.00]	3 [0.00]	0.25 [0.00]	4 [0.00]	0.27 [0.00]	3.80 [0.05]	2	200
Art5	0.36 [0.00]	2 [0.00]	0.20 [0.00]	4 [0.00]	0.36 [0.02]	3.36 [0.09]	9	900
Art6	0.53 [0.00]	1 [0.00]	0.34 [0.00]	2 [0.00]	0.53 [0.02]	1.68 [0.06]	4	400
Art7	0.54 [0.00]	4 [0.00]	0.72 [0.00]	4 [0.00]	0.66 [0.00]	3.68 [0.06]	1	100
Art8	0.61 [0.00]	5 [0.00]	0.76 [0.00]	5 [0.00]	0.66 [0.01]	3.74 [0.06]	1	1000
Iris	0.67 [0.00]	1 [0.00]	0.24 [0.00]	4 [0.00]	0.27 [0.01]	2.36 [0.08]	3	150
Wine	0.65 [0.00]	2 [0.00]	0.64 [0.00]	2 [0.00]	0.64 [0.00]	2.04 [0.04]	3	178
Glass	0.71 [0.00]	3 [0.00]	0.42 [0.00]	5 [0.00]	0.67 [0.01]	2.98 [0.07]	7	214
Pima	0.45 [0.00]	1 [0.00]	0.42 [0.00]	3 [0.00]	0.45 [0.00]	1.92 [0.11]	2	798
Soybean	0.15 [0.00]	3 [0.00]	0.08 [0.00]	4 [0.00]	0.10 [0.00]	3.90 [0.04]	4	47
Thyroid	0.38 [0.00]	3 [0.00]	0.24 [0.00]	3 [0.00]	0.36 [0.00]	2.76 [0.06]	3	215
CERIES	0.76 [0.00]	2 [0.00]	0.33 [0.00]	4 [0.00]	0.58 [0.02]	2.30 [0.11]	6	259

Table 2. Results obtained with 10-means and AntClass algorithms

database	10-Means		AntClass	
	Ec $[\sigma_{Ec}]$	K' $[\sigma_{K'}]$	Ec $[\sigma_{Ec}]$	K' $[\sigma_{K'}]$
Art1	0.18 [0.01]	8.58 [0.98]	0.15 [0.05]	4.22 [1.15]
Art2	0.38 [0.01]	8.52 [0.96]	0.41 [0.01]	12.32 [2.01]
Art3	0.31 [0.01]	8.28 [0.96]	0.35 [0.01]	14.66 [2.68]
Art4	0.32 [0.02]	6.38 [0.75]	0.29 [0.23]	1.68 [0.84]
Art5	0.08 [0.01]	8.82 [0.91]	0.08 [0.01]	11.36 [1.94]
Art6	0.10 [0.02]	8.46 [1.08]	0.11 [0.13]	3.74 [1.38]
Art7	0.87 [0.02]	7.76 [1.03]	0.17 [0.24]	1.38 [0.60]
Art8	0.88 [0.01]	8.78 [0.83]	0.92 [0.01]	13.06 [2.18]
Iris	0.18 [0.03]	7.12 [1.11]	0.19 [0.08]	3.52 [1.39]
wine	0.27 [0.01]	9.64 [0.52]	0.51 [0.11]	6.46 [2.10]
Glass	0.29 [0.02]	9.44 [0.70]	0.40 [0.06]	5.60 [2.01]
Pima	0.50 [0.01]	9.90 [0.36]	0.47 [0.02]	6.10 [1.84]
Soybean	0.13 [0.02]	8.82 [0.97]	0.54 [0.17]	1.60 [0.49]
Thyroid	0.42 [0.02]	9.56 [0.57]	0.22 [0.09]	5.84 [1.33]
CERIES	0.11 [0.01]	9.38 [0.63]	0.27 [0.15]	3.40 [1.06]

similarity can be used to sort the data either in increasing/decreasing order. With an increasing order, the first connected ants are those which are the less similar to all the others and therefore close to their cluster and far away from the others. With a decreasing order, the first ants to connect are the most similar to

all the others. Thus an ant belonging to a different cluster will have more chance to be connected than in the increasing case. The results obtained are given in table 1. It is obvious that the increasing order is more interesting regarding to the clustering error. We have consequently adopted this strategy in the following (with $L_{max} = 10$: at most 10 incoming links per ant).

We have compared AntTree with other clustering algorithms: AntClass [10, 11], a clustering algorithm inspired by a colony of artificial ants, and the Kmeans algorithm initialized with 10 randomly generated initial partitions (the data used for experimentation do not contain more than 10 clusters). Table 2 shows the results obtained for the 10-means and AntClass. We can see that AntTree gives an averaged error which is lower than AntClass for Art2, Art3, Art4, Art8, pima and soybean, and almost similar for Art1, glass, thyroid. Moreover, for the majority of the databases, the number of clusters found by AntTree is closer to the number of real classes than the number found by AntClass (10 databases out of 15).

AntTree is also better than the 10-means method for Art2, Art3, Art4, Art7, Art8, pima, soybean and thyroid. Moreover, the number of classes found by AntTree is also better (14 databases out of 15) for these second results. According to the standard deviations, we can also notice that AntTree is more precise than AntClass and 10-means.

Averaged computational time for one run is between 1 ms (thyroid database) and 0.5 s (Art3 composed of 1100 data). 10-means and AntClass were programmed in C, AntTree in C++ and the tests were performed on a standard PC (PIII 700 MHz). The averaged computational times of AntTree are relatively low compared to the two other methods because when an ant is connected to the tree, it will not move anymore. AntTree benefits from the fact that tree-based method often have a low complexity (like the heap sorting algorithm for instance).

5 Conclusion

In this paper we have described a new algorithm which is directly inspired from the ants self-assembly behavior. We have shown how it can be applied to the unsupervised learning problem and how it can obtain a tree-structured organization of the data. This tree can be either visualized in a hierarchical way or it can be interpreted as a flat partition. This method has been successfully compared with the Kmeans and AntClass, both in terms of clustering errors, number of classes and computational time. Those results are extremely encouraging and the main perspective of this work is to keep on studying this promising model.

More precisely, future work will deal with the unhooking capacity of ants (like drops of ants for the *L humiles* and the disaggregate of chains for *Oecophylla longinoda*). Each ant will have the possibility to disconnect itself from its position and to move on other ants that may be more similar. We are also interested in a probabilistic approach of our algorithm: for each ant we associate a probability of connecting which depends on its similarity with its close ants. We also want to

automatize the initialization and the updating of the thresholds. Finally we plan to apply this algorithm to the automatic hierarchical clustering of documents (Web pages) for the generation of portal sites.

References

1. Shin Ando and Hitoshi Iba. Ant algorithm for construction of evolutionary tree. In W. B. Langdon, editor, *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, page 131, New York, 9–13 July 2002. Morgan Kaufmann Publishers.
2. C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.
3. N-R Franks and A Sendova-Franks. Brood sorting by ants: distributing the workload over the work surface. *Behav. Ecol. Sociobiol*, 30:109–123, 1992.
4. S. Goss and J.-L. Deneubourg. Harvesting by a group of robots. In Varela, editor, *Proceedings of the First European Conference on Artificial Life*, pages 195–204, Sydney, Australia, 1991. Toward a Practice of Autonomous Systems.
5. C. Guinot, D. J.-M. Malvy, F. Morizot, M. Tenenhaus, J. Latreille, S. Lopez, E. Tschachler, and L. Dubertret. Classification of healthy human facial skin. Textbook of Cosmetic Dermatology Third edition (to appear), 2003.
6. A-K Jain and R-C Dubes. *Algorithms for Clustering Data*. Prentice Hall Advanced Reference Series, 1988.
7. P. Kuntz, D. Snyers, and P. Layzell. A stochastic heuristic for visualising graph clusters in a bi-dimensional space prior to partitioning. *Journal of Heuritics*, 5(3), October 1999.
8. Arnaud Lioni, Christian Sauwens, Guy Theraulaz, and J-L Deneubourg. The dynamics of chain formation in oecophylla longinoda. *Journal of Insect Behavior*, 14:679–696, 2001.
9. E.D. Lumer and B. Faieta. Diversity and adaptation in populations of clustering ants. pages 501–508, 1994.
10. N. Monmarché. On data clustering with artificial ants. In A.A. Freitas, editor, *AAAI-99 & GECCO-99 Workshop on Data Mining with Evolutionary Algorithms: Research Directions*, pages 23–26, Orlando, Florida, July 18 1999.
11. N. Monmarché, M. Slimane, and G. Venturini. On improving clustering in numerical databases with artificial ants. In D. Floreano, J.D. Nicoud, and F. Mondala, editors, *5th European Conference on Artificial Life (ECAL'99), Lecture Notes in Artificial Intelligence*, volume 1674, pages 626–635, Swiss Federal Institute of Technology, Lausanne, Switzerland, 13–17 September 1999. Springer-Verlag.
12. Guy Theraulaz, E Bonabeau, Christian Sauwens, J-L Deneubourg, Arnaud Lioni, F Libert, L Passera, and R-V Sol . Model of droplet formation and dynamics in the argentine ant (linepithema humile mayr). *Bulletin of Mathematical Biology*, 63:1079–1093, 2001.