Fuzzy Ant Based Clustering

Steven Schockaert, Martine De Cock, Chris Cornelis, and Etienne E. Kerre

Fuzziness and Uncertainty Modelling Research Unit
Department of Applied Mathematics and Computer Science
Ghent University
Krijgslaan 281 (S9), B–9000 Gent, Belgium
{steven.schockaert,martine.decock,chris.cornelis,etienne.kerre}@UGent.be
http://fuzzy.UGent.be

Abstract. Various clustering methods based on the behaviour of real ants have been proposed. In this paper, we develop a new algorithm in which the behaviour of the artificial ants is governed by fuzzy IF–THEN rules. Our algorithm is conceptually simple, robust and easy to use due to observed dataset independence of the parameter values involved.

1 Introduction

While the behaviour of individual ants is very primitive, the resulting behaviour on the colony-level can be quite complex. A particularly interesting example is the clustering of dead nestmates, as observed with several ant species under laboratory conditions [3]. Without negotiating about where to gather the corpses, ants manage to cluster all corpses into 1 or 2 piles. The conceptual simplicity of this phenomenon, together with the lack of centralized control and a priori information, are the main motivations for designing a clustering algorithm inspired by this behaviour. Real ants are, because of their very limited brain capacity, often assumed to reason only by means of rules of thumb [5]. Inspired by this observation, we propose a clustering method in which the desired behaviour of artificial ants (and more precisely, their stimuli for picking up and dropping items) is expressed flexibly by fuzzy IF–THEN rules.

The paper is organized as follows: in Section 2, we review existing work in the same direction, in particular the algorithm of Monmarché which served as our main source of inspiration. Section 3 familiarizes the reader with important notions about fuzzy set theory and fuzzy IF–THEN rules, while in Section 4 we outline the structure of our clustering algorithm and motivate its key design principles. Some experimental results are presented in Section 5. Finally, Section 6 offers some concluding remarks.

2 Related Work

Deneubourg *et al.* [3] proposed an agent–based model to explain the clustering behaviour of real ants. In this model, artificial ants (or agents) are moving randomly on a square grid of cells on which some items are scattered. Each cell

M. Dorigo et al. (Eds.): ANTS 2004, LNCS 3172, pp. 342–349, 2004.

can only contain a single item and each ant can move the items on the grid by picking up and dropping these items with a certain probability which depends on an estimation of the density of items of the same type in the neighbourhood.

Lumer and Faieta [8] extended the model of Deneubourg *et al.*, using a dissimilarity–based evaluation of the local density, in order to make it suitable for data clustering. Unfortunately, the resulting number of clusters is often too high and convergence is slow. Therefore, a number of modifications were proposed, by Lumer and Faieta themselves as well as by others (e.g. [4, 12]).

Monmarché [10] proposed an algorithm in which several items are allowed to be on the same cell. Each cell with a non–zero number of items corresponds to a cluster. Each (artificial) ant a is endowed with a certain capacity $c(a)$. Instead of carrying one item at a time, an ant a can carry a heap of $c(a)$ items. Probabilities for picking up at most $c(a)$ items from a heap and for dropping the load onto a heap are based on characteristics of the heap, such as the average dissimilarity between items of the heap. Monmarché proposes to apply this algorithm twice. The first time, the capacity of all ants is 1, which results in a high number of tight clusters. Subsequently the algorithm is repeated with the clusters of the first pass as atomic objects and ants with infinite capacity, to obtain a smaller number of large clusters. After each pass k–means clustering is applied for handling small classification errors.

In a similar way, in [6] an ant–based clustering algorithm is combined with the fuzzy c–means algorithm. Although some work has been done on combining fuzzy rules with ant-based algorithms for optimization problems [7], to our knowledge until now fuzzy IF–THEN rules have not yet been used to control the behaviour of artificial ants in a clustering algorithm.

3 Fuzzy IF–THEN Rules

A major asset of humans is their flexibility in dealing with imprecise, granular information; i.e. their ability to abstract from superfluous details and to concentrate instead on more abstract *concepts* (represented by words from natural language). One way to allow a machine to mimic such behaviour, is to construct an explicit interface between the abstract symbolic level (i.e. linguistic terms like "high", "old", ...) and an underlying, numerical representation that allows for efficient processing; this strategy lies at the heart of fuzzy set theory [13], which since its introduction in the sixties has rapidly acquired an immense popularity as a formalism for the representation of vague, linguistic information, and which in this paper we exploit as a convenient vehicle for constructing commonsense rules that guide the behaviour of artificial ants in our clustering algorithm.

Let us recall some basic definitions. A fuzzy set A in a universe U is a mapping from U to the unit interval $[0, 1]$. For any u in U, the number $A(u)$ is called the membership degree of u to A; it expresses to what extent the element u exhibits the property A. A fuzzy set R in $U \times V$ is also called a fuzzy relation from U to V. Fuzzy relations embody the principle that elements may be related to each other to a certain extent only. When $U = V$, R is also called a binary fuzzy relation in U. Classical set theory is tightly linked to boolean logic, in a sense that e.g. the operations of set complement, intersection and union are defined by means of logical negation, conjunction and disjunction respectively. This link is also maintained under the generalization from $\{0,1\}$ to $[0,1]$. For instance, to extend boolean conjunction, a wide class of operators called t–norms is at our disposal: a t–norm is any symmetric, associative, increasing $[0,1]^2 \rightarrow [0,1]$ mapping T satisfying $T(1,x) = x$ for every $x \in [0,1]$. Common t–norms include the minimum and the product in $[0,1]$, but also the Łukasiewicz t–norm T_W which has several desirable properties (see e.g. [11]) and which is defined by, for x, y in $[0,1]$,

$$T_W(x,y) = \max(0, x + y - 1) \tag{1}$$

Another prominent contribution of fuzzy set theory is the ability to perform *approximate reasoning.* In particular, we may summarize flexible, generic knowledge in a fuzzy rulebase like[1]

$$\begin{array}{c}\text{IF } X \text{ is } A_1 \text{ and } Y \text{ is } B_1 \text{ THEN } Z \text{ is } C_1\\ \text{IF } X \text{ is } A_2 \text{ and } Y \text{ is } B_2 \text{ THEN } Z \text{ is } C_2\\ \dots\\ \text{IF } X \text{ is } A_n \text{ and } Y \text{ is } B_n \text{ THEN } Z \text{ is } C_n\end{array}$$

where X, Y and Z are variables taking values in the respective universes U, V and W, and where for i in $\{1,\dots,n\}$, A_i (resp. B_i and C_i) is a fuzzy set in U (resp. V and W). Our aim is then to deduce a suitable conclusion about Z for every specific input of X and Y. Numerous approaches exist to implement this, with varying levels of sophistication; for our purposes, we used the conceptually simple and very efficient Mamdani method [9], that uses real numbers as inputs and outputs. It can be seen as a four–step process:

1. Given the observed values u of X and v of Y, we calculate for the i^{th} rule its activation level $\alpha_i = \min(A_i(u), B_i(v))$.
2. We "cut off" C_i at level α_i, i.e. we compute $C_i'(w) = \min(\alpha_i, C_i(w))$ for w in W. We thus obtain n individual conclusions.
3. The C_i''s are aggregated into the global inference result C' by means of $C'(w) = \max_{i=1}^{n} C_i'(w)$, for w in W.
4. Finally, a *defuzzification* method is used to transform the result into a crisp value of W; this can be, for instance, the center of gravity of the area below the mapping C' (center–of–gravity method, COG).

Another way of looking at Mamdani's method, is as a flexible way to interpolate an unknown, underlying (possibly very complex) mapping from $U \times V$ to W by means of linguistic labels.

[1] This can of course be generalized to an arbitrary number of variables in the antecedent.

4 Fuzzy Ants

Our algorithm is in many ways inspired by the algorithm of Monmarché [10]. We will consider however only one ant, since the use of multiple ants on a non–parallel implementation has no advantages[2]. Instead of introducing several passes, our ant can pick up one item from a heap or an entire heap. Which case applies is governed by a model of division of labour in social insects by Bonabeau *et al.* [2]. In this model, a certain stimulus and a response threshold value are associated with each task a (real) ant can perform. The response threshold value is fixed, but the stimulus can change and represents the need for the ant to perform the task. The probability that an ant starts performing a task with stimulus s and response threshold value θ is given by

$$T_n(s;\theta) = \frac{s^n}{s^n + \theta^n} \tag{2}$$

where n is a positive integer[3]. We will assume that $s \in [0,1]$ and $\theta \in]0,1]$.

Let us now apply this model to the problem at hand. A loaded ant can only perform one task: dropping its load. Let s_{drop} be the stimulus associated with this task and θ_{drop} the response threshold value. The probability of dropping the load is then given by

$$P_{drop} = T_{n_i}(s_{drop};\theta_{drop}) \tag{3}$$

where $i \in \{1,2\}$ and n_1, n_2 are positive integers. When the ant is only carrying one item n_1 is used, otherwise n_2 is used. An unloaded ant can perform two tasks: picking up one item and picking up all the items. Let s_{one} and s_{all} be the respective stimuli and θ_{one} and θ_{all} the respective response threshold values. The probabilities for picking up one item and picking up all the items are given by

$$P_{pickup\_one} = \frac{s_{one}}{s_{one} + s_{all}} \cdot T_{m_1}(s_{one};\theta_{one}) \tag{4}$$

$$P_{pickup\_all} = \frac{s_{all}}{s_{one} + s_{all}} \cdot T_{m_2}(s_{all};\theta_{all}) \tag{5}$$

where m_1 and m_2 are positive integers.

The values of the stimuli are calculated by evaluating fuzzy IF–THEN rules as explained below. We assume that the objects that have to be clustered belong to some set U, and that E is a binary fuzzy relation in U, which is reflexive (i.e. $E(u,u) = 1$, for all u in U), symmetric (i.e. $E(u,v) = E(v,u)$, for all u and v in U) and T_W–transitive (i.e. $T_W(E(u,v), E(v,w)) \leq E(u,w)$, for all u, v and w in U). For u and v in U, $E(u,v)$ denotes the degree of similarity between the

[2] Note, however, that the proposed changes do not exclude the use of multiple ants.

[3] In fact, this is a slight generalization which was also used in [12]; in [2] only the case were $n = 2$ is considered.

items u and v. For a non-empty heap $H \subseteq U$ with centre[4] c in U, we define the average and minimal similarity of H, respectively, by

$$avg(H) = \frac{1}{|H|} \sum_{h \in H} E(h, c) \qquad min(H) = \min_{h \in H} E(h, c) \tag{6}$$

Furthermore, let $E^*(H_1, H_2)$ be the similarity between the centres of the heap H_1 and the heap H_2.

4.1 Dropping Items

The stimulus for a loaded ant to drop its load L on a cell which already contains a heap H is based on the average similarity $A = avg(H)$ and an estimation of the average similarity between the centre of H and items of L. This estimation is calculated as $B = T_W(E^*(L, H), avg(L))$, which is a lower bound due to our assumption about the T_W–transitivity of E and can be implemented much more efficiently than the exact value. If B is smaller than A, the stimulus for dropping the load should be low; if B is greater than A, the stimulus should be high. Since heaps should be able to grow, we should also allow the load to be dropped when A is approximately equal to B. Our ant will perceive the values of A and B to be Very High (VH), High (H), Medium (M), Low (L) or Very Low (VL). The stimulus will be perceived as Very Very High (VVH), Very High (VH), High (H), Rather High (RH), Medium (M), Rather Low (RL), Low (L), Very Low (VL) or Very Very Low (VVL). These linguistic terms can be represented by fuzzy sets in $[0, 1]$. The rules for dropping the load L onto an existing heap H are summarized in Table 1(a).

Table 1. Fuzzy rules for inference of the stimulus for (a) dropping the load, (b) picking up a heap.

B \ A	VH	H	M	L	VL
VH	RH	H	VH	VVH	VVH
H	L	RH	H	VH	VVH
M	VVL	L	RH	H	VH
L	VVL	VVL	L	RH	H
VL	VVL	VVL	VVL	L	RH

(a)

B \ A	VH	H	M	L	VL
VH	VVH	-	-	-	-
H	M	VH	-	-	-
M	L	RL	H	-	-
L	VVL	VL	L	RH	-
VL	VVL	VVL	VVL	VL	M

(b)

[4] We do not go into detail about how to define and/or compute the centre of a heap, as this can be dependent on the kind of data that needs to be clustered.

4.2 Picking up Items

An unloaded ant should pick up the most dissimilar item from a heap if the similarity between this item and the centre of the heap is far less than the average similarity of the heap. This means that by taking the item away, the heap will become more homogeneous. An unloaded ant should only pick up an entire heap, if the heap is already homogeneous. Thus, the stimulus for an unloaded ant to pick up a single item from a heap H and the stimulus to pick up all items from that heap are based on the average similarity $A = avg(H)$ and the minimal similarity $M = min(H)$. The stimulus for picking up an entire heap, for example, can be inferred using the fuzzy rules in Table 1(b).

4.3 The Algorithm

During the execution of the algorithm, we maintain a list of all heaps. Initially there is a heap, consisting of a single element, for every item in the dataset. Picking up an entire heap H corresponds to removing a heap from the list. At each iteration our ant acts as follows. If the ant is unloaded, a heap from the list is chosen at random; the probabilities for picking up a single element and for picking up all elements are given by formulas (4)–(5). The case where H consists of only 1 or 2 items, should be treated separately (i.e. without using fuzzy rules). If the ant is loaded, a new heap containing the load L is added to the list of heaps with a fixed probability. Otherwise, a heap H from the list is chosen at random; the probability that H and L are merged is given by formula (3). The case where H consists of a single item, should be treated separately.

For evaluating the fuzzy rules, we used a Mamdani inference system with COG as defuzzification method. All response threshold values were set to 0.5. The other parameters are discussed in the next section.

5 Evaluation

We assume that the n objects to be clustered are characterized by m numerical attributes, i.e. $U = \{u_1, \ldots, u_n\}$ with $u_i \in \mathbb{R}^m$, $i = 1, \ldots, n$. To compute the similarity between vectors, we use the fuzzy relation E in U defined by, for u_i and u_j in U,

$$E(u_i, u_j) = 1 - \frac{d(u_i, u_j)}{d^*(U)} \tag{7}$$

where d represents Euclidean distance and $d^*(U)$ is (an estimation of) the maximal distance between objects from U. It can be proven that E is indeed reflexive, symmetric and T_W–transitive. To evaluate the algorithm, we compare the obtained clusters with the correct classification of the objects. For u in U, let $k(u)$ be the (unique) class that u belongs to and $c(u)$ the heap u was put in after algorithm execution. Following Monmarché [10], we define the classification error F_c by

$$F_c = \frac{1}{|U|^2} \sum_{1 \le i,j \le n} \epsilon_{ij} = \frac{2}{|U|(|U|-1)} \sum_{1 \le i < j \le n} \epsilon_{ij} \tag{8}$$

with

$$\epsilon_{ij} = \begin{cases} 0 & \text{if } (k(u_i) = k(u_j) \wedge c(u_i) = c(u_j)) \vee (k(u_i) \neq k(u_j) \wedge c(u_i) \neq c(u_j)) \\ 1 & \text{otherwise} \end{cases} \tag{9}$$

As an important benefit, this evaluation criterion strongly penalizes a wrong number of clusters [10]. As test cases for evaluating our algorithm, we took the "Wine", "Iris" and "Glass" datasets from the UCI Machine Learning Repository [1]. The "Wine" and "Iris" dataset consist of three classes; the "Glass" dataset consists of two main classes, each of which can be further split up into 3 subclasses. Table 2 shows the effect of changing the parameter n_1. For each dataset, the average value for F_c over 50 runs is shown; $(m_1, m_2, n_2) = (5, 5, 20)$ is kept constant. The results were obtained after 10^6 iterations of the algorithm. This reveals that $n_1 = 10$ is a good choice, and moreover small changes in the value of n_1 have little impact on the result. Similar conclusions can be drawn for the other parameter values[5]. The table also contains the results Monmarché obtained with his two–phase algorithm for these datasets. Clearly, for both "Wine" and "Glass" our results are a significant improvement. We also remark that the classification error of the "Glass" dataset was computed w.r.t. the 2 main classes, while Monmarché considered the 6 subclasses. Our algorithm always identifies the 2 main classes, while Monmarché's fails to identify either the 6 subclasses or the 2 main classes in a reliable way.

Initial experiments with artificial datasets suggest that for a dataset of size n, cn (with c an appropriate integer constant) is a good choice for the number of iterations. The corresponding execution time of the algorithm is approximately proportional to $n \log_2 n$, while rule evaluation happens in linear time.

Table 2. Influence of n_1 on F_c.

	$n_1 = 5$	$n_1 = 10$	$n_1 = 15$	$n_1 = 20$	Monmarché
Wine	0.50	0.13	0.14	0.16	0.51
Iris	0.17	0.16	0.17	0.17	0.19
Glass	0.16	0.12	0.13	0.14	0.40

6 Concluding Remarks

We have presented a clustering algorithm, inspired by the behaviour of real ants simulated by means of fuzzy IF-THEN rules. Like all ant-based clustering algorithms, no initial partitioning of the data is needed, nor should the number of clusters be known in advance. The machinery of approximate reasoning from fuzzy set theory endows the ants with some intelligence. As a result, throughout the whole clustering process, they are capable to decide for themselves to pick up either one item or a heap. Hence the two phases of Monmarché's original

[5] Due to limited space, we omit the corresponding data.

idea are smoothly merged into one, and k-means becomes superfluous. Initial experimental results with artificial datasets indicate good scalability to large datasets. Outliers in noisy data are left apart and hence do not influence the result, and the parameter values are observed to be dataset-independent which makes the algorithm robust.

Acknowledgments

Martine De Cock and Chris Cornelis would like to thank the Fund for Scientific Research – Flanders for funding their research.

References

1. Blake, C.L., Merz, C.J.: UCI Repository of Machine Learning Databases. University of California, available at: http://www.ics.uci.edu/~mlearn/MLRepository.html (1998)
2. Bonabeau, E., Sobkowski, A., Theraulaz, G., Deneubourg, J.L.: Adaptive Task Allocation Inspired by a Model of Division of Labor in Social Insects. *Working Paper* **98-01-004**, available at http://ideas.repec.org/p/wop/safiwp/98-01-004.html (1998).
3. Deneubourg, J.L., Goss, S., Franks, N., Sendova–Franks, A., Detrain, C., Chrétien, L.: The Dynamics of Collective Sorting Robot–Like Ants and Ant–Like Robots. From Animals to Animats: Proc. of the 1st Int. Conf. on Simulation of Adaptive Behaviour (1990) 356–363.
4. Handl, J., Meyer, B.: Improved Ant-Based Clustering and Sorting in a Document Retrieval Interface. Proc. of the 7th Int. Conf. on Parallel Problem Solving from Nature (2002) 913–923.
5. Hölldobler, B., Wilson, E.O.: The ants. Springer-Verlag Heidelberg (1990).
6. Kanade, P.M.,Hall, L.O.: Fuzzy Ants as a Clustering Concept. Proc. of the 22nd Int. Conf. of the North American Fuzzy Information Processing Soc. (2003) 227–232.
7. Lučić, P.: Modelling Transportation Systems using Concepts of Swarm Intelligence and Soft Computing. PhD thesis, Virginia Tech. (2002).
8. Lumer, E.D., Faieta, B.: Diversity and Adaptation in Populations of Clustering Ants. From Animals to Animats 3: Proc. of the 3th Int. Conf. on the Simulation of Adaptive Behaviour 501-508 (1994).
9. Mamdani, E.H., Assilian, S.: An Experiment in Linguistic Synthesis with a Fuzzy Logic Controller. Int. J. of Man-Machine Studies **7** (1975) 1–13.
10. Monmarché, N.: Algorithmes de Fourmis Artificielles: Applications à la Classification et à l'Optimisation, PhD thesis, Université François Rabelais (2000).
11. Klement, E.P., Mesiar, R., Pap, E.: Triangular norms, Kluwer Academic Publishers, Dordrecht (2002).
12. Ramos, V., Muge, F., Pina, P.: Self-Organized Data and Image Retrieval as a Consequence of Inter-Dynamic Synergistic Relationships in Artificial Ant Colonies. Soft Computing Systems: Design, Management and Applications (2002) 500-509.
13. Zadeh, L.A.: Fuzzy sets. Information and Control **8** (1965) 338–353.