A first approach to study the effects of direct information exchange between agents in ant-based clustering

Marco A. Montes de Oca, Leonardo Garrido and José L. Aguirre
Centro de Sistemas Inteligentes
Tecnológico de Monterrey
Eugenio Garza Sada 2501, Col. Tecnológico, Monterrey, N.L. México C.P. 64849
Email: {A00788072, leonardo.garrido, jlaguirre}@itesm.mx

***Abstract*— This paper presents a first approach to study direct information exchange between agents in ant-based clustering by comparing the clustering process development generated by agents that maintain a short-term memory against the one generated by agents that share their memory with their peers whenever an encounter occurs on the environment. The information exchange can allow an agent to "change its mind" regarding the most favorable dropping location on the grid, based on the knowledge of another agent.**

Our experimental evidence shows that this simple information exchange strategy improves the quality of the resultant clustering. This holds true, however, only for a small number of agents. This suggests that there is a critical number of agents that can exchange information, that when surpassed, the effects could even be detrimental.

I. Introduction

Insect species are considered eusocial when overlapping generations of parents and offspring live together in an organizational unit or colony, division of labor occurs within the colony, and when they develop a physical structure or nest to live in. Ants, termites and some bee and wasp species are considered eusocial [1].

In eusocial insects, different activities are performed simultaneously by specialized individuals. This division of labor can be seen in activities such as nest cleaning and brood care. For example, when dead bodies of nestmates and defeated enemies are scattered around the nest entrance of some ant species such as *Pogonomyrmex badius* and *Lasius niger*, worker ants create piles of their corpses on a distant spot away from it [2], [3]. Another example is brood care which is a highly specialized activity that require, in some cases, the separation of larvae according to their development stage. In experiments, the ant *Leptothorax unifasciatus* gather its larvae together according to their size. Small larvae are aggregated near the center of a plate and larger larvae near the periphery [3].

All these collective behaviors and many others exhibited by social insects are the outcome of a process of self-organization [4]. Bonabeau et al. [3] define self-organization as "a set of dynamical mechanisms whereby structures appear at the global level of a system from interactions among its lower-level components" (p. 9). They also identify four basic elements of self-organization, which are positive feedback, negative feedback, the amplification of fluctuations, and the existence of multiple interactions. In the cases of ant corpse piling and brood sorting there is a factor that mediates the self-organization process. This factor is known as *stigmergy*. First proposed by Grassé (cited in [4]) to explain the construction of nests by the termites *Cubitermes* and *Macrotermes*. Grassé observed that when workers of *Macrotermes bellicosus* were placed in a container with some soil pellets, the insects carried about and put down pellets in an apparently random fashion after an exploration phase where they moved through the container without taking any action. At this stage, a pellet just put down by a termite worker is often picked up and placed somewhere else by another worker. When a pellet is placed on top of another, the resultant structure appears to be much more attractive and termites soon start to pile more pellets nearby, making the dropping spot even more attractive [2]. Stigmergy then, is the indirect influence on the behavior of others through local environment modifications.

Inspired by corpse aggregation, brood sorting and nest building in colonies of social insects, computer scientists have created clustering algorithms for exploratory data analysis where insects are simulated by simple reactive agents that act in a two-dimensional grid. The basic clustering algorithm makes use of stigmergy as the only mean of communication between agents [5], [6].

Nevertheless, stigmergy is not the only way social insects interact with each other. In most species, trophallaxis or liquid food exchange among members of the same colony, plays a key role in their social organization [2]. Consider the case of some termite species which require intestinal protozoa to derive benefits from cellulose. Their early instar nymphs are fed either by oral or anal trophallaxis. The latter infects them with symbiotic protozoa or bacteria contained in the proctodeal liquid. The subsocial association result of this codependence have evolved into a complex social and morphological structure [1].

Inspired by the oral trophallaxis phenomenon, we present a first approach to study the effects of direct information exchange between agents in ant-based clustering algorithms.

We compare the clustering process development generated by the Lumer and Faieta [6] algorithm that uses a short-term memory with an extension of it where information exchange between agents is possible. We provide experimental evidence that show the advantages of direct information exchange in the quality of the clustering obtained.

The remaining of this paper is organized as follows. In section II previous work on ant-based clustering using stigmergy as the only mean of communication between agents is presented. Section III describes the model of information exchange used. Section IV details the experimental strategy used to measure the effects of introducing information exchange into the basic ant-based clustering algorithm. Section V presents the experiments results. In section VI we analyze the experiments results and discuss the implications of the introduction of information exchange bewteen agents in ant-based clustering algorithms. Finally, in section VII we conclude and speculate on future extensions and ideas to study the impact of information exchange in ant-based clustering algorithms.

II. Previous Work

A. Spatial sorting

Ant-based clustering was introduced by Deneubourg et al. [5] using a model for spatial sorting. A group of agents exhibiting the same behavior move randomly over a toroidal square grid. In the environment there are objects that were initially scattered in a random fashion. The objects can be picked up, moved or dropped in any free location on the grid. An object is picked up with high probability if it is not surrounded by other objects of the same type and is dropped by a loaded agent if its neighborhood is populated by other objects of the same type and the location of the agent has no object on it. The probability p_p with which an object is picked up is given by

$$p_p = \left(\frac{k_1}{k_1 + f}\right)^2 \tag{1}$$

where k_1 is a constant and f is considered to be a measure of similarity of the agent's neighborhood with the object at the agent's current location. If the object is surrounded by objects of the same type, then f should be large in comparison with k_1 in order to make p_p small. On the other hand, if the object is not surrounded by any object of the same type, then f should be small making p_p high. The probability p_d for an agent to deposit an object is given by

$$p_d = \left(\frac{f}{f + k_2}\right)^2 \tag{2}$$

where k_2 is another constant. If the agent's neighborhood is populated by objects of the same type as the object it is carrying, then f should be large in comparison with k_2 in order to make p_d approach 1. On the other hand, if the agent is not surrounded by any object similar to its load, then f should be small making p_d small. As expected, p_d has an opposite behavior than p_p.

Having a robotic implementation in mind, Deneubourg et al. [5] proposed to compute f through a short-term memory each agent maintains. Keeping track of the last T time units, an agent counts the number of objects it finds; f is the number of objects of a particular type encountered by the agent in the last T time steps divided by the maximum number of objects an agent can find in T time units.

B. Exploratory Data Analysis

Exploratory data analysis tries to identify structures and relationships within data and cluster analysis is one possible approach to do so. In cluster analysis, the main problem is that of finding a partition of a data set so that similar objects end up in the same class or cluster [7].

Lumer and Faieta [6] generalized the spatial sorting algorithm to apply it to exploratory data analysis specifically, through the creation of clusters of related data. All classical clustering algorithms depend on a similarity or dissimilarity measure to determine whether two objects are similar or not. This algorithm is no exception. However, instead of taking as input a similarity or dissimilarity matrix, this algorithm starts with something very similar to what we described in the preceding section. A group of agents moving randomly on a toroidal square grid. On the grid, data elements scattered randomly that can be picked up, moved or dropped by the agents. This time however, the probabilities are computed differently. The probability of picking a data element i is defined as

$$p_p(i) = \left(\frac{k_p}{k_p + f(i)}\right)^2 \tag{3}$$

where k_p is a constant and $f(i)$ is a similarity density measure with respect to element i. Likewise, the probability of dropping a data element is given by

$$p_d(i) = \begin{cases} 2f(i) & \text{if } f(i) < k_d \\ 1 & \text{otherwise} \end{cases} \tag{4}$$

where k_d is a constant. The similarity density $f(i)$ for an element i, at a particular grid location τ, is defined as

$$f(i) = \begin{cases} \frac{1}{s^2} \sum_{j \in Neigh(\tau)} \left(1 - \frac{d(i,j)}{\alpha}\right) & \text{if } f > 0 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where s^2 is the size of the perception area centered at the location of the agent and α is a scaling factor of the dissimilarity measure $d(i, j)$ between elements i and j. If it is assumed that the elements can be represented as points in an n-dimensional space, then the Euclidean distance could be used as dissimilarity measure.

In experiments, Lumer and Faieta [6] showed that the algorithm was capable of correctly classifying elements of a synthetic data set; however, the number of clusters found was, in general, greater than the number expected. To fix this problem, they proposed three modifications to the basic algorithm.

The first modification was the introduction of variability into the agent population. The element of change was the velocity

of displacement. With this modification, velocities are in the range $[1, V_{max}]$ and represent the number of grid positions the agent can move in a single time unit along a given direction vector. This variability was coupled to the sensitiveness of each agent to dissimilarity in the following way

$$f(i) = \begin{cases} \frac{1}{s^2} \sum_{j \in Neigh(\tau)} \left(1 - \frac{d(i,j)}{\alpha\left(1+\frac{v-1}{V_{max}}\right)}\right) & \text{if } f > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

The result is that an agent moving slowly will be more selective in their similarity criterion than an agent moving rapidly. The effect of this variability is that the number of final clusters is closer to the expected than the model with homogeneous agents. Fast moving agents appear to create clusters in a coarser scale than slowly moving agents which place data elements more carefully.

The second modification was the introduction of a short-term memory into the agents so that they could keep track of the dropping location of the last m elements the agent had carried. This modification has the effect of lowering the probability that a just picked up element can create an independent cluster by itself, thus reducing the final number of clusters. This happens because every time an agent picks an element, it looks in its memory for the location of the closest element in attribute space. The agent, instead of moving randomly, moves directly towards that particular location.

The third modification was the inclusion of behavioral switches. If an agent has not manipulated an item for a predetermined number of time units, it starts destroying the so far created clusters. This does not end up in a total cluster destruction because other agents, that have not reached that state, will reposition the disseminated objects.

C. Clustering Validation

In an effort to formally evaluate the clustering quality of ant-based algorithms, Handl et al. [8] applied four validity measures to the resulting clustering of a modified version of the Lumer and Faieta [6] algorithm. Two of them are *external* validity coefficients, meaning that they rely on a pre-defined structure which is imposed on a data set and reflects our intuition about the clustering structure of that data set. The other two coefficients are *internal* validity measures. In this case the clustering results are evaluated using only the data vectors available in the data set. For the interested reader, the modifications introduced by Handl et al. can be found in [9].

The four validity measures used by Handl et al. [8] were :

- External validity coefficients. Let $\mathbf{C} = C_1, C_2, \ldots, C_m$ be a clustering structure of a given data set X and $\mathbf{P} = P_1, P_2, \ldots, P_n$ a defined partition of X. A pair $(\mathbf{x}_v, \mathbf{x}_u) \in X \times X$ is refered as
  - **SS**. If both points belong to the same cluster in $\mathbf{C}$ and to the same partition in $\mathbf{P}$.
  - **SD**. If both points belong to the same cluster in $\mathbf{C}$ but to different partitions in $\mathbf{P}$.
  - **DS**. If both points belong to different clusters in $\mathbf{C}$ but to the same partition in $\mathbf{P}$.
  - **DD**. If both points belong to different clusters in $\mathbf{C}$ and to different partitions in $\mathbf{P}$.

  If a, b, c and d are the number of pairs SS, SD, DS and DD respectively, then $a+b+c+d = M = N(N-1)/2$, that is, the maximum number of possible distinct pairs. The two external validity coefficients used are:
  1) The harmonic mean of recall and precision, also known as the *F-Measure*. Commonly associated to the information retrieval field, recall and precision are measures that give us some idea of how well a clustering algorithm is identifying the classes present in a database [10].
     Let $rec(i,j)$ be the recall of cluster j with respect to partition i. Then, $rec(i,j) = |C_j \bigcap P_i|/|P_i|$. The precision of cluster j with respect to partition i is defined as $pre(i,j) = |C_j \bigcap P_i|/|C_j|$. The F-measure is defined as
     $$F_{i,j} = \frac{2\, pre(i,j)\, rec(i,j)}{pre(i,j) + rec(i,j)}$$
     The overall F-Measure for a given clustering is
     $$F = \sum_{i=1}^{m} \frac{|P_i|}{N} max_{j=1,\ldots,n}\{F_{i,j}\} \tag{7}$$
     The F-Measure takes the value of 1 when a perfect match between the clustering and the given partition occurs.
  2) The Rand Statistic. It is defined by
     $$R = \frac{a+d}{a+b+c+d} \tag{8}$$
     The Rand statistic is limited to the interval $[0, 1]$ with a value of 1 with a perfect clustering.
- Internal validity coefficients.
  1) The intra-cluster variance. This measure tries to capture the idea that members that belong to the same cluster should be as close to each other as possible. It is given by
     $$I = \sum_{i=1}^{n} \sum_{p \in C_i} \|p - \mu_i\|^2 \tag{9}$$
     where n is the total number of groups in the partition generated by the clustering algorithm, and μ_i is the centroid of group i. This measure is to be minimized.
  2) The Dunn index. This measure will give a high value whenever the partition gives compact and well separated clusters. It is given by
     $$D = \min_{c,d \in C} \left\{ \frac{\|\mu_c - \mu_d\|}{\max_{e \in C}\{diam(e)\}} \right\} \tag{10}$$
     where μ_i is the centroid of cluster i and $diam(i)$ is the diameter of cluster i which could be considered a dispersion measure. It is defined as
     $$diam(c) = \max_{x,y \in c}\{\|x - y\|\}$$

In this investigation we used these validity measures to evaluate the effects of information exchange between agents during the clustering process.

III. Information Exchange

In all previous attempts to do clustering tasks using a simulated insect colony there is no direct interaction among agents. This is perhaps due to the fact that early works on ant-based clustering and sorting were focused on robotic implementations [4], [5], [11], [12] where direct real-time communication is much more complicated than in software simulations.

As a first approach to study the effects of direct information exchange between agents in ant-based clustering, we introduce a modification to the model of clustering ant proposed by Lumer and Faieta [6] that uses a short-term memory to bias an agent's search for a dropping location. In the original model, every agent has a memory in which it stores the m most recent elements dropped along with their locations. When a new element is picked up, it is compared with the elements in memory. After that, the ant moves directly towards the location of the most similar element in its memory.

The model used in our approach uses as analogy the oral trophallaxis phenomenon among nestmates in colonies of social insects. Whenever a group of agents coincide on the grid, a loaded agent of that group exchanges knowledge with its peers about the environment state. In general, the main purpose of this exchange is to better approximate the current environment state in order to bias an agent search for better dropping locations. In other words, a loaded agent can "change its mind" more than once regarding the most favorable dropping location, based on the knowledge of other agents.

The knowledge exchange is done by allowing an agent look into its peers short-term memories for a closer object to its load than the object that was guiding it on its search. If it finds it, then it redirects itself towards the location of that object.

IV. Experimental Setup

A. Test Data

To evaluate the impact of the introduction of basic information exchange between agents we used two real data collections from the UCI Machine Learning Repository [13]. These were :

- Iris Plant Database. Composed of 150 instances with 4 numeric attributes each. There are 3 classes in the database composed of 50 instances each. Of these, one is linearly separable from the other two; the latter are not linearly separable from each other.
- Wine Recognition Database. Composed of 178 instances with 17 numeric attributes each. There are 3 classes in the database where class 1 has 59 instances, class 2 has 71 instances, and class 3 has 48 instances.

To eliminate the bias on similarity measures provoked by different scales within data attributes, we standardized all data sets. The similarity measure used in all our experiments was the cosine metric because it is commonly used when dealing with real databases and gave better results than Euclidean distance in preliminary tests.

B. Ant Model

The agents picking and dropping probabilities were computed using expresions 3 and 4 respectively. However, because the cosine metric is a similarity measure, expresion 5 is not applicable. Therefore, for the local similarity density $f(i)$, we used

$$f(i) = \frac{1}{s^2} \sum_{j \in Neigh(\tau)} \left(\frac{1}{1 + e^{-S \frac{d(i,j)}{\alpha} + D}} \right) \qquad (11)$$

where S is the steepness of the response curve and D serves as a displacement factor. In our experiments we fixed S to 5 because it provides a similarity value close to 0 when the cosine measure is minimum, that is, when the cosine mesure gives a value of -1, and D to 1 because this allows us to better distinguish vectors with separation angles between 0 and $\pi/2$. The resultant similarity response surface is shown in figure 1.

Equation 11 permits the integration of simple background knowledge. In particular, if it is known that data classes are difficult to separate, one can move the displacement factor near the maximum value of the similarity measure to better discriminate among very similar data elements. This expression also has the advantage of limiting the range of values α can take to $(0, 1]$, given that $-1 \leq d(i, j) \leq 1$, as is the case for the cosine metric.

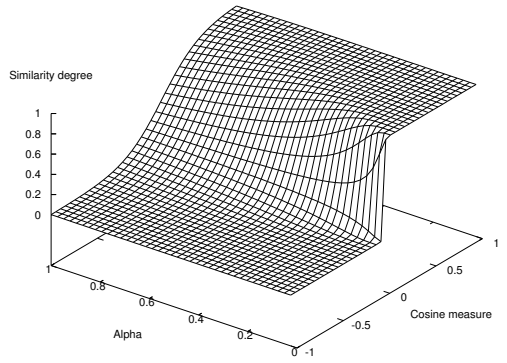


Fig. 1. Similarity response surface. Similarity response defined as a function of the cosine metric between two data elements and the scaling factor α. Taken from the similarity component of equation 11 where $S = 5$ and $D = 1$.

Table I summarizes all other agent settings used in our experiments. These settings were selected because they proved acceptable with both databases.

C. Experiments Design

To observe the effects of information exchange between agents on the clustering process, we applied the four measures used by Handl et al. [8] and mentioned in subsection II-C every 10000 simulation cycles to the partial clustering hitherto obtained. Each simulation cycle was composed of N individual actions, where N is the number of agents in the

TABLE I
AGENT SETTINGS

Parameter	Value
k_p	0.1
k_d	0.15
α	0.7
Neighborhood size	5×5
Short-term memory size	8

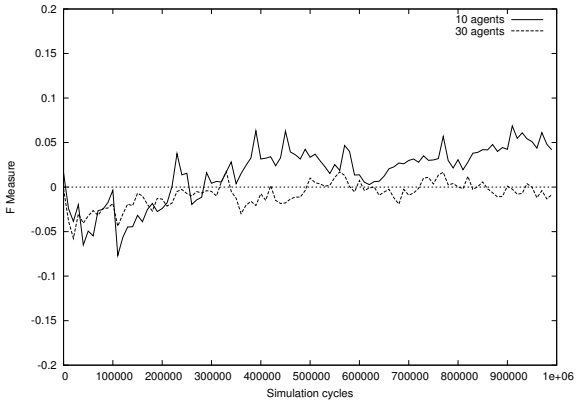
simulation. We ran both, Lumer and Faieta algorithm and ours 30 times with every database for 1000000 simulation cycles.

In our experiments, information exchange only occurs whenever two or more agents meet at a point on the grid. We can expect therefore, that the probability of an encounter raises as the number of agents is increased, or more precisely, as the agents density of the environment increases. The agents density can be modified either by changing the number of agents or changing the environment dimensions. We chose to change only the number of agents because changing the environment dimensions would also result in a modification of the data density which would have introduced another factor to take into account. By having more agent encounters over time, the effects of information exchange should have more impact. We tried with populations of 10, 20 and 30 agents within an environment of 100×100 locations in all of our experiments. Considering we tried with databases with an average of 160 elements each, incrementing the number of agents beyond 30 would have provoked scarcity of data elements on the grid and would have reduced artificially the number of clusters found, thus making the task of discovering patterns on data more difficult.

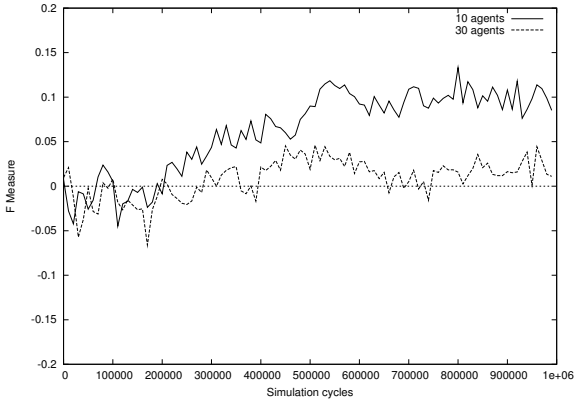
To collect information from the spatial partition formed by the agents on the grid, we applied an agglomerative hierarchical algorithm on the data using Euclidean distance and the single linkage strategy. The maximum distance to merge clusters was set equal to the size of the agents neighborhood. The distance used to compute the variance and the clusters diameters in the attribute space was also the Euclidean distance.

V. RESULTS

Results of the Lumer and Faieta algorithm and our modified version on the test databases are summarized and presented grouped by validity measure. In order to better observe the performance difference between the two algorithms when varying the number of agents, the plots show the resulting curve of substracting the evaluations for the algorithm without communication from the evaluations for our version. Subsection V-E presents the results of the number of clusters found. Results with 20 agents are skipped because, in all cases, they showed a behavior in between the results with 10 and 30 agents.



(a) Iris database



(b) Wine database

Fig. 2. Relative F-Measure scores over time for the short-term memory algorithm with information exchange on the Iris Plant and the Wine databases using 10 and 30 agents. The 0 value serves as reference and corresponds to the evaluations for the algorithm without information exchange.

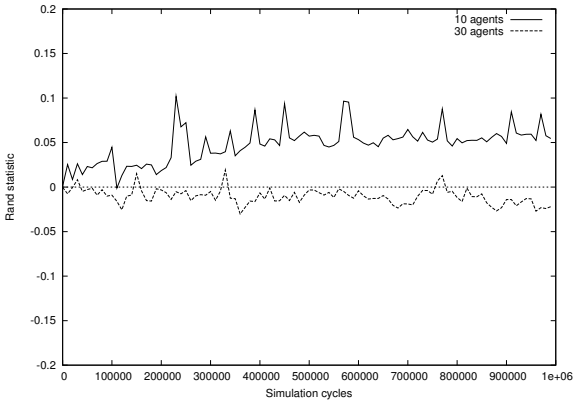
A. F-Measure

Figure 2 shows the relative F-Measure values over time for both algorithms on the Iris Plant and Wine databases using 10 and 30 agents. In both cases, it can be seen that the algorithm with information exchange has a higher score than the algorithm without information exchange after a certain number of simulation cycles. Moreover, the relative difference between the algorithms when increasing the number of agents is evident. The relative F-Measure score is better when using 10 agents.

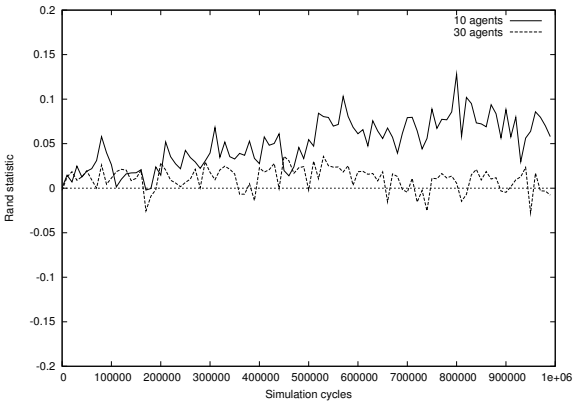
B. Rand Statistic

The relative Rand statistic scores over time for both algorithms on the Iris Plant and Wine databases using 10 and 30 agents are shown in figure 3.

In figure 3 almost the same situation than that of the F-Measure can be observed, i.e., with 30 agents the performance is worse than with only 10 agents and than the performance of its counterpart without information exchange. This can be seen

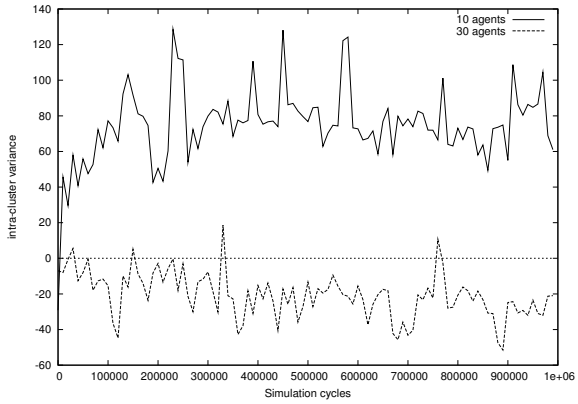


(a) Iris database

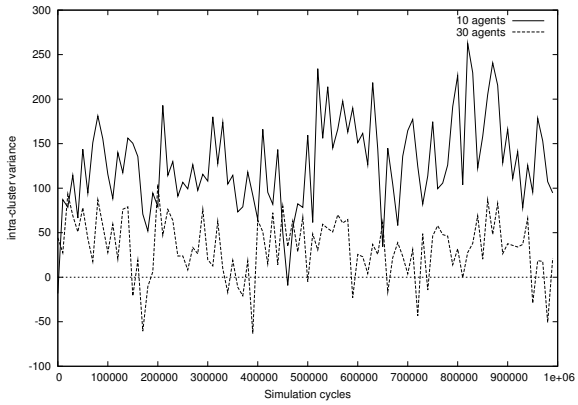


(b) Wine database

Fig. 3. Relative Rand statistic scores over time for the short-term memory algorithm with information exchange on the Iris Plant and the Wine databases using 10 and 30 agents. The 0 value serves as reference and corresponds to the evaluations for the algorithm without information exchange.



(a) Iris database



(b) Wine database

Fig. 4. Relative total intra-cluster variance over time of the short-term memory algorithm with and without information exchange on the Iris Plant and the Wine databases using 10 and 30 agents. The 0 value serves as reference and corresponds to the evaluations for the algorithm without information exchange.

in subfigure 3(a) where the line corresponding to the algorithm with information exchange using 30 agents is below the zero line.

C. Intra-cluster Variance

In order to be consistent with figures 2 and 3, i.e., values above the zero line meaning good performance, in figure 4 the curves were obtained by substracting the intra-cluster variance obtained by the algorithm version with information exchange from the intra-cluster variance obtained by the algorithm without information exchange.

Figure 4 shows how the total intra-cluster variance gap between the two versions gets smaller as the number of agents increases. In the Iris database (fig. 4(a)), the curves even swap positions making the information exchange version with 30 agents the worst.

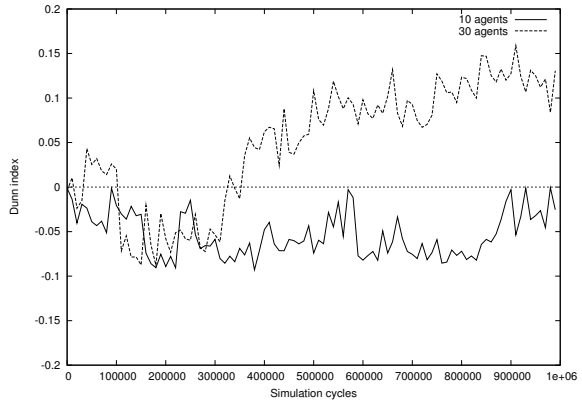
D. Dunn index

Figure 5 shows the relative Dunn index scores for both algorithms with 10 and 30 agents on the two test databases. In this case, the performance of the short-term memory algorithm without information exchange is better in the Iris database and worse in the Wine database compared with the information exchange algorithm using 10 agents.

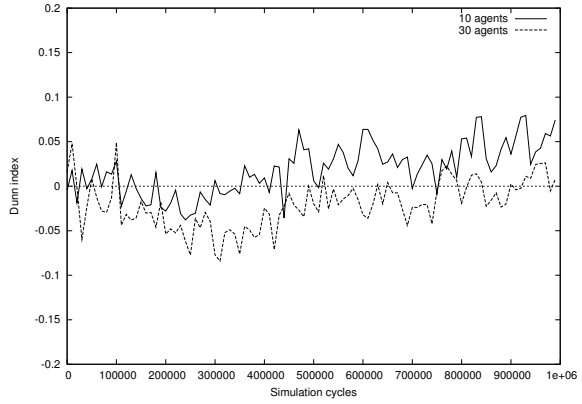
Figure 5 also shows that the performance of the information exchange algorithm using 30 agents is better in the Iris database and almost the same in the Wine database compared with the short-term memory algorithm. This behavior can be explained if we recall that in the Iris database two clusters are not linearly separable, thus making the algorithm consider them as one. This clearly increases the measure that tries to identify compact and well separated clusters.

E. Number of clusters

One of the main features that make ant-based clustering an appealing approach to clustering is that it is capable, on average, of finding the correct number of clusters within the data. This is confirmed in figures 6 and 7.

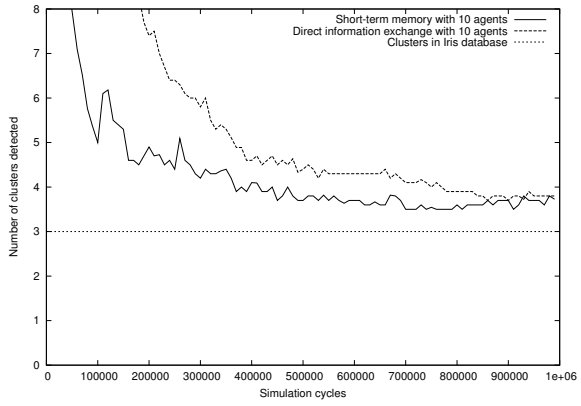


(a) Iris database

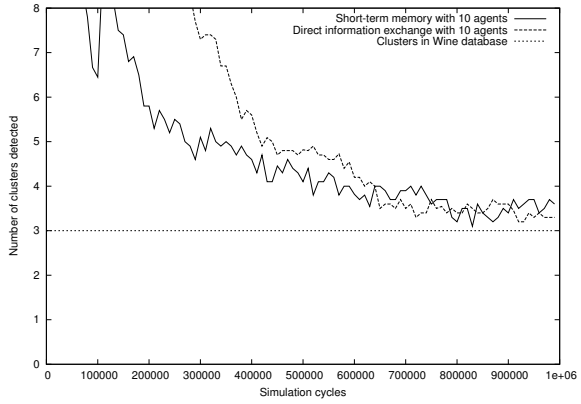


(b) Wine database

Fig. 5. Relative Dunn index scores over time for the short-term memory algorithm with and without information exchange on the Iris Plant and the Wine databases using 10 and 30 agents. The 0 value serves as reference and corresponds to the evaluations for the algorithm without information exchange.



(a) Iris database



(b) Wine database

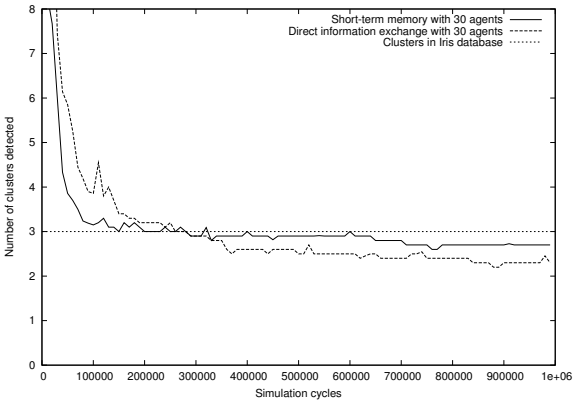
Fig. 6. Number of clusters found by the short-term memory algorithm with and without information exchange on the Iris Plant and the Wine databases using 10 agents. As a reference, the correct number of clusters is shown.

Figure 7 shows how the number of clusters decreases as the number of agents increases. The information exchange version has a slower decay rate.

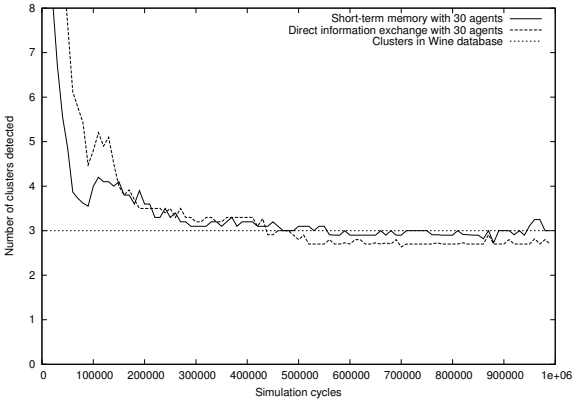
VI. Discussion

The probability of an encounter between two agents moving randomly raises as the number of agents is increased. This is confirmed in robotic experiments [4] where it has been reported that the number of collisions between robots raises as the number of robots is increased. In our experiments, we tried to take advantage of this fact and use it to study the effect of increasing the information exchange rate between agents. However, the actual effect of information exchange depends not only on the frequency of encounters but also on their effectiveness; that is, information exchange is beneficial as long as the information exchanged is *useful*. In our approach, usefulness depends very much on how well an agent's memory represents the *current* environment state. Early in the clustering process this is not accomplished because of the high environment dynamism. When this phase finishes, once that some initial clusters exist, information exchange is more effective as the environment is less dynamic. This in part explains the relatively poor performance of the information exchange model over the first simulation cycles.

Our results suggest that even by just sharing memorized dropping spots, information exchange offers some advantages over the model that relies on stigmergy as the only mean of interaction among agents. In particular, the quality of the clustering generated is improved and this is reflected by the F-Measure, Rand index and intra-cluster variance scores. This holds true, however, only when the number of agents is small. This fact suggests that there is a critical number of agents that can exchange information, that when surpassed, the effects could even be detrimental. Despite this, information exchange could be useful when one cannot afford the computational costs associated with increasing the number of agents in a clustering task.



(a) Iris database



(b) Wine database

Fig. 7. Number of clusters found by the short-term memory algorithm with and without information exchange on the Iris Plant and the Wine databases using 30 agents. As a reference, the correct number of clusters is shown.

VII. Conclusions and Future Work

The information exchange schema explored in this paper is just an extension of the Lumer and Faieta [6] ant model that provides an agent with a short-term memory to bias its search for a dropping location. The extension consists basically on allowing the agents to explore each other's memories whenever they meet at a point on the grid. The result is that when an agent is in search for a dropping location, it can change its trajectory based on the knowledge of other agents.

There are many possibilities to explore direct communication among agents in ant-based clustering. In all of them one has to determine *what* information to exchange; *how* to exchange this information, and *when* is most appropriate to do so. Some of this aspects have been investigated in the robotics field, and although their results are important, they are not directly aplicable to software implementations where physical restrictions need not be considered. The potential benefits of direct communication to ant-based clustering are important in a time where the volume of available electronic information is overwhelming and when organizations need to extract useful knowledge from it.

Let us emphasize that the study of direct communication in ant-based custering algorithms does not mean the abandonment of the swarm intelligence paradigm because direct communication do happen in the biological inspiration model, i.e., insects colonies.

Our future efforts will be focused on exploring different information exchange strategies that, while keeping the agents simple, could provide improvements in the quality of the resulting clustering of ant-based clustering algorithms. Among these, the exploration of indirect information exchange through deposition of packets of information on the environment inspired by anal trophallaxis, and the exchange of "maps" that are built while an agent is moving around the environment.

Acknowledgment

The authors would like to thank Julia Handl for clarifying some aspects of her work. To César Marín and Eduardo H. Ramírez for their comments about early drafts of this paper.

This research has been supported by the Technologies for Distributed Knowledge and Intelligent Agents Research Programme CAT-011. Tecnológico de Monterrey, Campus Monterrey.

References

[1] R. J. Elzinga, *Fundamentals of entomology*. Prentice Hall, 2000.

[2] E. O. Wilson, *The Insect Societies*. The Belkap Press of Harvard University Press, 1971.

[3] E. Bonabeau, M. Dorigo, and G. Theralauz, *Swarm Intelligence. From Natural to Artificial Systems*. Oxford University Press, 1999.

[4] O. Holland and C. Melhuish, "Stigmergy, self-organization, and sorting in collective robotics," *Artificial Life*, vol. 5, pp. 173–202, 1999.

[5] J.-L. Deneubourg, S.Goss, N. Franks, A. Sendova-Franks, C. Detrain, and L. Chretien, "The dynamics of collective sorting: Robot-like ants and ant-like robots," in *Proceedings of the First International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, pp. 356–365, MIT Press, 1991.

[6] E. D. Lumer and B. Faieta, "Diversity and adaptation in populations of clustering ants," in *Proceedings of the Third International Conference on Simulation of Adaptive Behavior: From Animals to Animats 3*, pp. 501–508, MIT Press, 1994.

[7] A. D. Gordon, *Classification*. Monographs on statistics and applied probability 82, Chapman & Hall/CRC, second ed., 1999.

[8] J. Handl, J. Knowles, and M. Dorigo, "On the performance of ant-based clustering." Design and application of hybrid intelligent systems. Frontiers in Artificial Intelligence and Applications 104. IOS Press, Amsterdam, The Netherlands, 2003.

[9] J. Handl, J. Knowles, and M. Dorigo, "Ant-based clustering: a comparative study of its relative performance with respect to k-means, average link and 1d-som," Tech. Rep. TR/IRIDIA/2003-24, IRIDIA, Universite Libre de Bruxelles, Belgium, 2003.

[10] S. M. zu Eissen and B. Stein, "Analysis of clustering algorithms for web-based search," in *Practical Aspects of Knowledge Management LNAI 2569*, pp. 168–178, 2002.

[11] A. Martinoli and F. Mondada, "Collective and cooperative group behaviours: Biologically inspired experiments in robotics," in *Proceedings of the Fourth International Symposium on Experimental Robotics ISER-95* (O. Khatib and J. Salisbury, eds.), pp. 3–10, Springer Verlag, 1995.

[12] C. R. Kube and H. Zhang, "Collective robotic intelligence," in *Second International Conference on Simulation of Adaptive Behavior*, pp. 460–468, 1992.

[13] C. Blake and C. Merz, "UCI repository of machine learning databases [http://www.ics.uci.edu/∼mlearn/mlrepository.html]," 1998.