Using Entropy for Evaluating Swarm Intelligence Algorithms

Gianluigi Folino and Agostino Forestiero

Abstract. In the last few years, the bio-inspired community has experienced a growing interest in the field of Swarm Intelligence algorithms applied to real world problems. In spite of the large number of algorithms using this approach, a few methodologies exist for evaluating the properties of self-organizing and the effectiveness in using these kinds of algorithm. This paper presents an entropy-based model that can be used to evaluate self-organizing properties of Swarm Intelligence algorithms and its application to SPARROW-SNN, an adaptive flocking algorithm used for performing approximate clustering. Preliminary experiments, performed on a synthetic and a real-world data set confirm the presence of self-organizing characteristics differently from the classical flocking algorithm.

1 Introduction

Swarm Intelligence (SI) [1] is an innovative computational method for solving problems that originally took its inspiration from the biological examples provided by social insects such as ants, termites, bees, etc. These systems are typically made up of a population of simple agents interacting directly or indirectly (by acting on their local environment) with each other. Indirect interaction, i.e. when an individual modify the environment and the other responds to this change, is named *stigmergy* [5]. This mechanism permits to reduce direct communication among agents, and must be taken into account when designing artificial systems. In practice, an agent deposits something in the environment that makes no direct contribution to the task being undertaken, but is used to influence the subsequent behavior that is task related. Although there is normally no centralized control structure dictating how individual agents should behave, local interactions between such agents often lead to the emergence of global behavior. Examples of systems like these can be found in nature, including ant colonies, bird flocking, animal herding, bacteria molding

Gianluigi Folino · Agostino Forestiero
Institute for High Performance Computing and Networking (ICAR-CNR)
e-mail: {folino,forestiero}@icar.cnr.it

J.R. González et al. (Eds.): NICSO 2010, SCI 284, pp. 331–343, 2010.
springerlink.com

and fish schooling. The advantages of SI are twofold. Firstly, it offers intrinsically distributed algorithms that can use parallel computation quite easily. Secondly, the use of multiple agents supplies a high level of robustness, as the failure of a few individuals does not alter too much the behavior of the overall system.

Clustering is the act of partitioning an unlabeled dataset into groups of similar objects. Each group, called a cluster, consists of objects that are similar between themselves and dissimilar to objects of other groups.

The SPARROW-SNN (Shared Nearest-Neighbor similarity), better described in [4], couples an adaptive flocking algorithm with a shared nearest neighbor (SNN) [3] cluster algorithm to discover clusters with differing sizes, shapes in noise and high dimensional data.

In the last few years, innovative algorithms based on SI models [7] [8][10][2] have been introduced to solve real world problems in a decentralized fashion (i.e. the clustering problem, correlated to the SPARROW-SNN algorithm).

In spite of the large number of algorithms using this approach, a few methodologies exist for evaluating the properties of self-organizing and the effectiveness of using these kinds of algorithm. In this work, a methodology based on the concept of entropy inspired by the paper [11], is illustrated. The main principle stated in the paper is that the key to reduce disorder in a multi-agent system and to achieve a coherent global behavior is coupling that system to another in which disorder increases. This corresponds to a macro-level where the order increases, i.e. a coherent behavior arises, and a micro-level where an increase in disorder is the cause for this coherent behavior at the macro-level.

Using this approach, the self organizing properties of SI algorithms can be experimentally evaluated, considering macro and micro levels of entropy. The method is applied to evaluate self-organizing properties of the SPARROW-SNN, the adaptive flocking algorithm cited above. However, it can be easily applied to any type of SI algorithm.

The rest of this paper is organized as follows: Section 2 presents the multi agent adaptive flocking algorithm for searching interesting objects and shows how this algorithm can be used as a basis for clustering spatial data, combining it with a local merging strategy based on the SNN algorithm; Section 3 introduces a methodology based on the concept of entropy useful to evaluate self-organizing properties of SI based algorithms. Section 4 shows how entropy can be experimentally evaluated and used to assess the goodness of the flocking algorithm.

Finally, section 5 draws some conclusions.

2 An Adaptive Flocking Algorithm

In this section, a multi-agent adaptive flocking algorithm is presented, which has the advantage of being easily implementable on parallel and distributed machines and is robust compared to the failure of individual agents. First, the rules governing the flock model originally introduced by Reynolds [13] are explained; then, the modified behavioral rules of the swarm agents are illustrated. They add an adaptive behavior to the flock and make it more effective in searching points, which have some desired properties in the space.

The flocking algorithm was proposed by Reynolds as a method for simulating the flocking behavior of birds on a computer both for animation and as a way to study emergent behavior. Flocking is an example of emergent collective behavior: there is no leader, i.e., no global control. Flocking behavior emerges from the local interactions. In the flock algorithm each agent has direct access to the geometric description of the whole scene, but reacts only to flock mates within a certain small radius. The basic flocking model consists of three kind of simple steering behavior:

Separation gives an agent the ability to maintain a certain distance from others nearby. This prevents agents from crowding too closely together, allowing them to scan a wider area.

Cohesion supplies an agent with the ability to cohere (approach and form a group) with other nearby agents. Steering for cohesion can be computed by finding all agents in the local neighborhood and computing the average position of the nearby agents. The steering force is then applied in the direction of that average position.

Alignment gives an agent the ability to align with other nearby characters. Steering for alignment can be computed by finding all agents in the local neighborhood and averaging together the 'heading' vectors of the nearby agents.

Our flocking algorithm extends Reynolds's rules and is inspired by a work presented by Macgill [9], first introducing colored agents.

The algorithm starts with a fixed number of agents that occupy a randomly generated position in the search space. Each agent moves around the spatial data, testing the neighborhood of each location in order to verify whether a point can have some desired properties. Each agent follows the rules of movement described in Reynolds' model. In addition, this model considers four different kinds of agents, classified on the basis of some properties of data in their neighborhood. Different agents are characterized by a different color: red, revealing interesting patterns in the data, green, a medium one, yellow, a low one, and white, indicating a total absence of patterns.

The main idea behind this approach is to take advantage of the colored agent in order to explore more accurately the most interesting regions (signaled by the red agents) and avoid the ones without interesting properties (signaled by the white agents). Red and white agents stop moving in order to signal this type of region to the others, while green and yellow ones fly to find denser zones. Indeed, each flying agent computes its heading by taking the weighted average of alignment, separation and cohesion (as illustrated in figure 1).

The following are the main features which make this model different from Reynolds' model:

- *Alignment* and *cohesion* do not consider yellow agents, since they move in a not very attractive zone.
- *Cohesion* is the resultant of heading towards the average position of the green flockmates (centroid), of the attraction towards reds, and of the repulsion from whites, as illustrated in figure 1.
- A *separation* distance is maintained from all the agents, apart from their color.

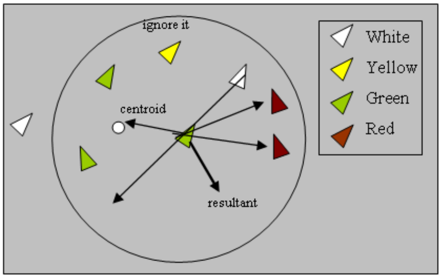


Fig. 1 Computing the direction of a green agent

2.1 Formal Description of the Flock

Consider the search space, in which the swarm moves, having dimension d. Let N be the number of birds and B be the set of all the birds $\{B_1, B_2, \ldots, B_N\}$. Each bird B_k can be represented by three d-dimensional vectors: its position in this space $Pos_k : (x_k^1, x_k^2, \ldots, x_k^d)$, its direction $Dir_k : (dir_k^1, dir_k^2, \ldots, dir_k^d)$, where dir_k^i represents the component along the axis i of the direction of the bird and the color $Col_k \in \{white, yellow, green, red\}$, indicating the type of bird. We used as distance between two birds B_a and B_b, the euclidean distance between their respective positions: $dist(B_a, B_b) = \sqrt{\sum_{i=1}^{d}(x_a^i - x_b^i)^2}$.

We define as $dist\_max$ and $dist\_min$ respectively the radius indicating the limited sight of the birds and the minimum distance that must be maintained among them. $Neigh(B_k)$ denotes the neighborhood of a bird B_k, i.e. the set $\{B_\alpha \in B \mid dist(B_k, B_\alpha) \leq dist\_max\}$, that is the set of the birds visible from the bird B_k. Furthermore, we define as $Neigh(col, B_k)$ the set $\{B_\alpha \in B \mid dist(B_k, B_\alpha) \leq dist\_max, Col_\alpha = col\}$, that is the set of the birds, having color col, visible from the bird B_k.

Each bird moves with speed v, depending on the color of the agents (green agents' speed is slower, because they are exploring interesting zones). Then, for each iteration t, the new position of a bird B_k can be computed as:

$$\forall i = 1 \ldots d \qquad x_k^i(t+1) = x_k^i(t) + v \times dir_k^i \tag{1}$$

Note also that for each iteration the new direction of the agent k is obtained summing the three components of alignment, separation and cohesion:

$$dir_k^i = dir\_al_k^i - dir\_sep_k^i + dir\_co_k^i. \tag{2}$$

Considering as $dir(B_a, B_b)$ the normalized direction of the vector between a bird B_a and a bird B_b, these components can be computed using the following formulas (3, 4 and 5):

$$dir\_al_k^i = \frac{1}{|Neigh(green, B_k)|} \cdot \sum_{B_\alpha \in Neigh(green, B_k)} dir_\alpha^i \tag{3}$$

and considering $centr(green, B_k)$ as the position of the centroid of the green agents in the neighborhood of k with generic coordinate i:
$\frac{1}{|Neigh(green,B_k)|} \cdot \sum_{B_\alpha \in Neigh(green,B_k)} x_\alpha^i$, then:

$$dir\_co_k^i = dir(centr(green, B_k), B_k)^i + attr\_red - rep\_white \tag{4}$$

where $attr\_red$ is equals to:

$$\sum_{B_\alpha \in Neigh(red, B_k)} dir(B_\alpha, B_k)^i$$

and rep_white is equals to:

$$\sum_{B_\alpha \in Neigh(white, B_k)} dir(B_\alpha, B_k)^i$$

i.e the sum of the attraction towards the centroid, of the attraction towards the red birds and of the repulsion from the white birds;

$$dir\_sep_k^i = \sum_{B_\alpha \in Neigh(B_k), dist(B_\alpha, B_k) < dist\_min} dir(B_\alpha, B_k)^i \tag{5}$$

2.2 Using the Flocking Algorithm for Clustering Spatial Data

SNN is a clustering algorithm developed by Ertöz, Steinbach and Kumar [3] to discover clusters with differing sizes, shapes and densities in noise and high dimensional data. The algorithm extends the nearest-neighbor non-hierarchical clustering technique by Jarvis-Patrick [6] redefining the similarity between pairs of points in terms of how many nearest neighbors the two points share. Using this new definition of similarity, the algorithm eliminates noise and outliers, identifies representative points, and then builds clusters around the representative points. These clusters do not contain all the points, but rather represent relatively uniform group of points.

SPARROW-SNN combine the strategy of search of the previously described clustering algorithm with the SNN algorithm main principles for discovering clusters of arbitrary form and density. In practice, the flocking algorithm performs a biased sampling of the points of the dataset, as it focuses the search on interesting parts of the search space. Thus, the SNN algorithm was applied to merge the clusters and to eliminate the noise points. A more complete description of the algorithm can be found in [4].

To better understand as SPARROW-SNN works, the pseudocode was shown in figure 2. The algorithm starts with a fixed number of agents placed in a randomly generated position. From their initial position, each agent moves around the spatial

```
for i=1 ... MaxIterations
    foreach agent (yellow, green)
        age=age+1;
        if (age > Max_Life)
            generate_new_agent();die();
        endif
        if (not visited (current_point))
            property = compute_local_property(current_point);
            mycolor= color_agent(property);
        endif
    end foreach
    foreach agent (yellow, green)
        dir= compute_dir();
    end foreach
    foreach agent (all)
        switch (mycolor){
          case yellow, green: move(dir, speed(mycolor)); break;
          case white: stop(); generate_new_agent(); break;
          case red: stop(); generate_new_close_agent(); break; }
    end foreach
end for
```

Fig. 2 The pseudo-code of the adaptive flocking algorithm

data testing the neighborhood of each location in order to verify whether the point can be identified as a *representative* (or core) point.

The *compute_property* function represents the connectivity of the point as defined in the SNN algorithm. In practice, when an agent falls on a data point A, not yet visited, it computes the connectivity, *conn(A)*, of the point, i.e. computes the total number of strong links the points has, according to the rules of the SNN algorithm. Points having connectivity smaller than a fixed threshold (*noise_threshold*) are classified as noise and are considered for removal from clustering. Then a color is assigned to each agent, on the basis of the value of the connectivity computed in the visited point, using the following procedure (called *color_agent*() in the pseudocode):

$$\begin{array}{lll} conn > core\_threshold & \Rightarrow mycolor = red & (speed = 0) \\ noise\_threshold < conn \leq core\_threshold & \Rightarrow mycolor = green & (speed = 1) \\ 0 < conn < noise\_threshold & \Rightarrow mycolor = yellow & (speed = 2) \\ conn = 0 & \Rightarrow mycolor = white & (speed = 0) \end{array}$$

The colors assigned to the agents are: red, revealing representative points, green, border points, yellow, noise points, and white, indicating an obstacle (uninteresting region). After the coloration step, the green and yellow agents compute their movement observing the positions of all other agents that are at most at some fixed distance (*dist_max*) from them and applying the rules described in the previous subsection. In any case, each new red agent (placed on a representative point) will run the merge procedure, so that it will include, in the final cluster, the representative point discovered, together to the points that share with them a significant (greater that P_min) number of neighbors and that are not noise points. The merging phase considers two different cases: when points in the neighborhood have never been visited and when there are points belonging to different clusters. In the former, the same temporary label will be assigned and a new cluster will be constituted; in the latter, all the points will be merged into the same cluster, i.e. they will get the label corresponding to the smallest label. Thus clusters will be built incrementally.

3 An Entropy-Based Model

This section describes the application of a new methodology for understanding and evaluating self-organizing properties in bio-inspired systems. The approach is experimentally evaluated on the flocking system of the previous subsection, but it could be easily applied to any bio-inspired systems.

To this aim, we used a model based on the entropy introduced in [11] by Parunak and Brueckner. The authors adopted a measure of entropy to analyze emergence in multi-agent systems. Their fundamental claim is that the relation between self-organization based on emergence in multi-agent systems and concepts as entropy is not just a loose metaphor, but it can provide quantitative and analytical guidelines for designing and operating agent systems. These concepts can be applied in measuring the behavior of multi-agent systems. The main result, that the above cited paper suggests, concerns the principle that the key to reduce disorder in a multi-agent system and to achieve a coherent global behavior is coupling that system to another in which disorder increases. This corresponds to a macro-level where the order increases, i.e. a coherent behavior arises, and a micro-level where an increase in disorder is the cause for this coherent behavior at the macro-level.

A multi-agent system follows the second law of thermodynamics “Energy spontaneously disperses from being localized to becoming spread out if it is not hindered”, if agents move without any constriction. However, if we add information in an intelligent way, the agents’ natural tendency to maximum entropy will be contrasted and the system will go towards self-organization. For the sake of simplicity, in the case of the flocking, the attractive behavior of the red birds and the repulsive effect of the white agents add self-organization to the system, while in the case of ant systems, this is typically originated from the pheromone.

Really, as stated in [11], we can observe two levels of entropy: a macro level in which organization takes place, balanced by a micro level in that we have an increase of entropy. For the sake of clarity, in the flocking algorithm, micro-level is represented by red and white agents’ positions, signaling respectively interesting and desert zones, and the macro level is computed considering all the agents’ positions. So, we expect to observe an increase in micro entropy due to the birth of new red and white agents and, on the contrary, a decrease in macro entropy indicating organization in the coordination model of the agents.

[12] defines autocatalytic property for agent systems as follows: "A set of agents has autocatalytic potential if, in some regions of their joint state space, their interaction causes system entropy to decrease (and thus leads to increased organization). In that region of state space, they are autocatalytic". As for our algorithm, at the beginning, the agents move and spread out randomly. Afterward, the red agents act as catalyzers towards the most interesting zones, organization increases and entropy should decrease. Note that in the case of ants, attraction is produced by the effect of pheromone, while for our flock, it is caused by the attractive power of the red birds (and by the repulsion of the white birds).

Now, a more formal description of the entropy-based model is described. In information theory, entropy can be defined as:

$$S = -\sum_{i} p_i \log p_i \tag{6}$$

Now, to adapt this formula to our aims, a location-based (*locational*) entropy is introduced. Consider an agent moving in a space of data divided in a grid $N \times M = K$, where all the cells have the same dimensions. So, if N and M are quite large and each agent is placed in a randomly chosen cell of this grid (as in the first iteration of the flocking algorithm), then the probability that the agent is in one of the K cells of the grid is equal for all the agents.

The entropy can be measured experimentally running the flocking algorithm for T tries and counting how many times an agent falls in the same cell i for each time-step. Dividing this number by T we obtain the probability p_i that the agent be in this cell.

Then, the locational entropy will be:

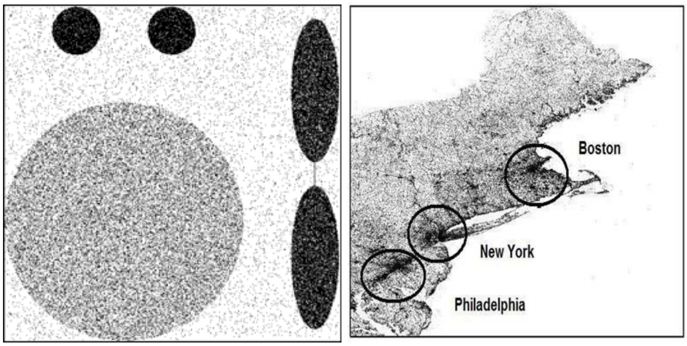
$$S = -\frac{\sum_{i=1}^{k} p_i \log p_i}{\log k} \tag{7}$$

In the case of a random distribution of the agents, every state has probability $\frac{1}{k}$, so the overall entropy will be $\frac{\log k}{\log k} = 1$; this explains the factor of normalization log k in the formula. Obviously, in the case of the flocking algorithm, clustering zones will be visited more frequently and the probability will be higher in this zones and lower outside them. Consequentially, the entropy will be lower than 1. This situation can be verified for the Cure dataset (figure 3 a), observing the probability distribution (figure 4) in the grid.

The above equation can be generalized for P agents, summing over all the agents and averaging dividing by P. Equation (7) represents the *macro-entropy*; if we consider only red and white agents, it represents the *micro entropy*.

4 Experimental Results

Using the approach described in the previous section, the micro and macro entropy has been evaluated experimentally. All the experiment have been conducted using the real world North-East dataset, showed in figure 3 b, containing 123,593 points representing postal addresses of three metropolitan areas (New York, Boston and Philadelphia) in the North East States, It comprises a lot of noise represented from distributed rural areas and smaller towns. The artificial Cure dataset (figure 3 a) is also used, as it presents a cluster distribution quite regular and this permits a better understanding of the catalytical properties; in fact, the dataset contains 100,000 points distributed in three circles and two ellipsoids and connected by a chain of outliers and random noise scattered in the entire space.



(a) (b)

Fig. 3 a) CURE dataset. b) North-East dataset

We run the adaptive flocking algorithm (averaged over 100 tries) for 2000 time-steps using 100 agents and the same standard parameters for the flocking algorithm of the work in [4] and computed the probability an agent falls in every cell of the grid (as shown in figure 4 for the CURE dataset). Using these data and settings, we computed the micro and macro locational entropy both for SPARROW-SNN and, for the sake of comparison, for the random search algorithm and for the classical Reynolds model (without the adaptive behavior of the colored birds).

The result of these experiments, for the North-East dataset, is reported in figure 5. As expected, we can observe an increase in micro entropy (figure 5 b) and a decrease in macro entropy (figure 5 a) due to the organization introduced in the coordination model of the agents by the attraction towards red agents and the repulsion of white agents. On the contrary, in random search and standard flock model, the curve of macro entropy is almost constant, confirming the absence of organization.

A similar trend was observed for the Cure dataset (here not reported for the lack of space).

In addition, we conducted simulations in order to verify the property of autocatalysm of our system and to better understand the behavior of our algorithm specifically for the CURE dataset.

In figures 6 a and b, respectively the entropy in the cluster zones and outsides the clusters is reported. Entropy decreases both in cluster zones and outsides the

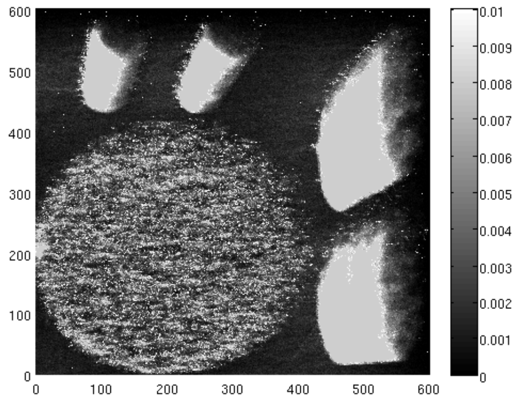


Fig. 4 Probability that an agent falls in a cell of the Grid for the Cure dataset (probability greater than 0.01 is set to 0.01)

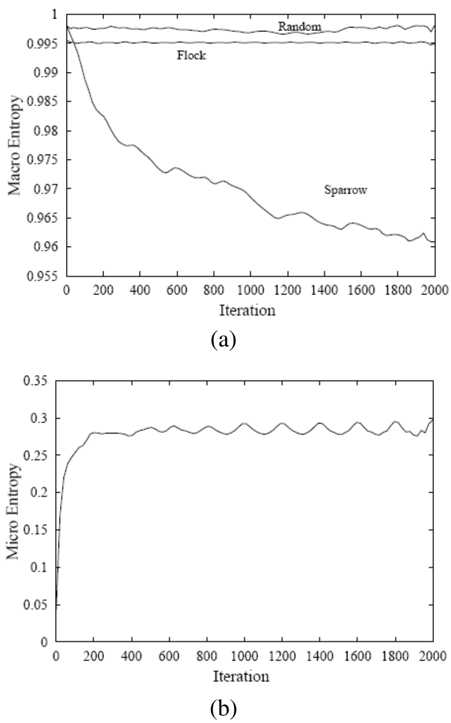
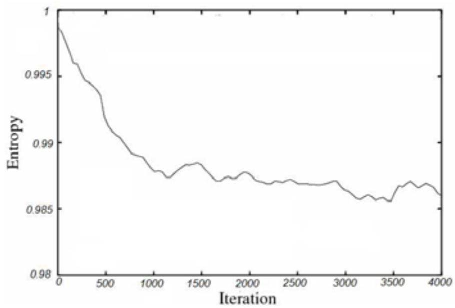
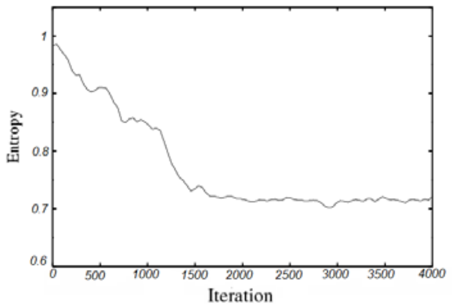


Fig. 5 North-East dataset: a) Macro Entropy (all the agents) using SPARROW-SNN, random search and standard flock b) Micro Entropy (red and white agents) using SPARROW-SNN



(a)



(b)

Fig. 6 Macro Entropy a) inside the cluster zones and b) outside the cluster zones for the Cure dataset using SPARROW-SNN, random and standard flock

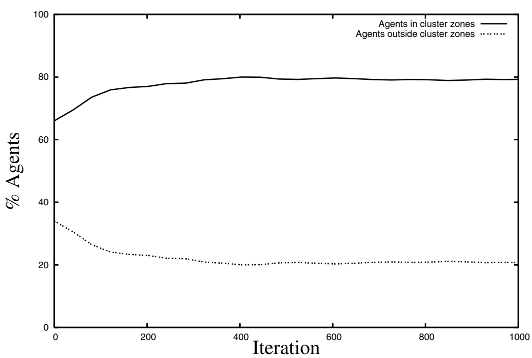


Fig. 7 Percentage of agents exploring cluster and non cluster zones for the Cure dataset using SPARROW-SNN

clusters zones, as the flock visits more frequently cluster zones and keeps away from the other zones (this behavior also causes a decrease in the entropy).

However these curves are not sufficient to verify the effectiveness of the algorithm as organization alone is not sufficient to solve problems, but it must bring the search in the appropriate zones. In fact, the main idea behind our algorithm is to let the flock explore the search space and, when the birds reach a desirable region (zone dense of clusters), an autocatalytic force is applied to the system (red birds) to keep searching in these zones.

Thus, we analyzed the average percentage of birds present in these two different zones (figure 7). In cluster zones we have about the 80% of the entire flock (while the space occupied by the clusters is about 65%) and this confirms the goodness of the algorithm, as in the interesting zones of the clusters, not only there is organization but there also a larger presence of searching agents.

5 Conclusions

This paper shows how an entropy-based model can be used to evaluate self-organizing properties of SI algorithms. Preliminary experiments, conducted using a flocking algorithm successfully employed for performing approximate clustering, demonstrate the presence of self-organizing characteristics differently from random search and classical flocking algorithm. However, entropy alone is not sufficient to assess the goodness of the algorithm in searching the space (i.e. performing clustering) and other measures are needed in order to verify the search is concentrated in interesting zones. Anyway, we believe that this model could be useful to better understand and control the behavior of multi-agent systems and to drive the user for choosing the appropriate parameters. Future works aim to evaluate and compare self-organization properties of other SI models, as Ants Colony Optimization, Particle Swarm Optimization, etc..

References

[1] Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm intelligence: from natural to artificial systems. Oxford University Press, New York (1999)
[2] Ellabib, I., Calamai, P., Basir, O.: Exchange strategies for multiple ant colony system. Inf. Sci. 177(5), 1248–1264 (2007), http://dx.doi.org/10.1016/j.ins.2006.09.016
[3] Ertoz, L., Steinbach, M., Kumar, V.: A new shared nearest neighbor clustering algorithm and its applications. In: Workshop on Clustering High Dimensional Data and its Applications at 2nd SIAM International Conference on Data Mining, pp. 105–115 (2002)
[4] Folino, G., Forestiero, A., Spezzano, G.: An adaptive flocking algorithm for performing approximate clustering. Inf. Sci. 179(18), 3059–3078 (2009)
[5] Grassé, P.: La Reconstruction du nid et les Coordinations Inter-Individuelles chez Beellicositermes Natalensis et Cubitermes sp. La Théorie de la Stigmergie: Essai d'interprétation du Comportement des Termites Constructeurs in Insect. Soc. 6. Morgan Kaufmann, San Francisco (1959)
[6] Jarvis, R.A., Patrick, E.A.: Clustering using a similarity measure based on shared nearest neighbors. IEEE Transactions on Computers C 22(11) (1973)

[7] Kuntz, P., Snyers, D.: Emergent colonization and graph partitioning. In: SAB 1994: Proceedings of the third international conference on Simulation of adaptive behavior: from animals to animats, vol. 3, pp. 494–500. MIT Press, Cambridge (1994)
[8] Lumer, E.D., Faieta, B.: Diversity and adaptation in populations of clustering ants. In: From Animals to Animats: Proc. of the Third Int. Conf. on Simulation of Adaptive Behaviour, pp. 501–508. MIT Press, Cambridge (1994)
[9] Macgill, J.: Using flocks to drive a geographical analysis engine. In: Bedau, M.A., McCaskill, J.S., Packard, N.H., Rasmussen, S. (eds.) Artificial Life VII: Proceedings of the Seventh International Conference on Artificial Life, pp. 446–453. The MIT Press, Cambridge (2000)
[10] Monmarché, N., Slimane, M., Venturini, G.: On improving clustering in numerical databases with artificial ants. In: Floreano, D., Mondada, F. (eds.) ECAL 1999. LNCS, vol. 1674, pp. 626–635. Springer, Heidelberg (1999)
[11] Parunak, H.V.D., Brueckner, S.: Entropy and self-organization in multi-agent systems. In: AGENTS 2001: Proceedings of the fifth international conference on Autonomous agents, pp. 124–130. ACM Press, New York (2001), `http://doi.acm.org/10.1145/375735.376024`
[12] Parunak, H.V.D., Brueckner, S.: Engineering swarming systems. In: Methodologies and Software Engineering for Agent Systems, pp. 341–376. Kluwer, Dordrecht (2004)
[13] Reynolds, C.W.: Flocks, herds and schools: A distributed behavioral model. In: SIGGRAPH 1987: Proceedings of the 14th annual conference on Computer graphics and interactive techniques, pp. 25–34. ACM Press, New York (1987), `http://doi.acm.org/10.1145/37401.37406`