Proceedings of Student/Faculty Research Day, CSIS, Pace University, May 7th, 2004

# Ant Colonies, Self-Organizing Maps, and A Hybrid Classification Model

Michael L. Gargano, Lorraine L. Lurie, Lixin Tao, and Sung-Hyuk Cha

**Abstract**

Ant Colonies (or Swarm Intelligence) and Self-Organizing Maps (a type of unsupervised Neural Network) have become two important and powerful classification heuristics in computer science and artificial intelligence. After first describing each model, a hybrid is introduced that has the visual appeal of swarm intelligence and the efficiency of self-organizing maps.

**Keywords:** Neural Networks, Swarm Intelligence, Classification

## Ant Colonies (Swarm Intelligence)

Ants, bees, termites, and wasps are classified as social insects because they live in colonies. Every individual in a social insect colony seems to act independently of the others, but yet the colony functions as an organized unit. These social colonies can be thought of as natural problem solving systems (e.g., finding food, division of labor, building nests, and responding to a changing environment) having collective intelligence. Swarm intelligence emerges through the interactions among individual agents and also between the agents and their environment [1, 2, 3]. Since problem solving is an important aspect of computer science and related disciplines, these models are proving to be a powerful metaphor for artificially intelligent systems.

Agent based systems are appealing to computer scientists because of the emergence of collective intelligence through simple individual agent actions. The use of agent-based systems allows complex behaviors and self-organization to emerge from small, simple, interacting entities. In social insect colonies, the whole is more than the sum of its parts. The advantages of this exciting new paradigm are that it is robust, autonomous, flexible, adaptive, lends itself to distributed and parallel computation, and obtains optimal/near optimal solutions to problems having high complexity. Many successful applications have been designed and successfully implemented in the areas of classification, combinatorial optimization, communications networks, and robotics. Self-organization is the goal of swarm intelligent systems. In order for self-organization to emerge, the artificial colonies must emulate several important features. *Positive feedback*, or amplification, is one such important feature. It can be implemented through the reinforcement of a possible solution or through the recruitment of other agents into a particular activity. The next necessary feature is *negative feedback*. Negative feedback helps to stabilize the system. It can take the form of saturation, exhaustion, or competition. *Randomness* is also important to this model. This allows the system to move away from local optima. The final requirement is *multiple interactions*. This includes the individual knowledge of an agent and the knowledge of the actions of others. Obtaining the knowledge of other agents is accomplished through direct interaction with agents as well as indirect interaction, which is communicated by changes in the environment made by others. The latter type is also called **stigmergy**.

In self-organized systems there seems to be an integration of individual activities that does not require any supervision or overall structure. Cooperation among the agents in the colony is largely self-organized. The self-organized structure of the agents is mostly due to communication between them, whether direct or stigmergic. Individual actions might be simple (for example, one ant following the pheromone trail of another) but collectively they can solve problems such as finding optimal paths to food sources for the entire nest. This collective behavior is called **swarm intelligence**. For example, routing traffic in telecommunication networks can be viewed as an ant foraging problem. In some situations, this ant foraging behavior can also be used to allocate tasks in an efficient and cost effective manor. In our example, analyzing large amounts of data can be treated as a clustering problem simulating ant larvae sorting and dead body clustering as models for their solution.

Social insect larvae areas are complex and intricate structures. Stigmergy, that is, the coordination of agent activities through the environment, is an important mechanism underlying the successful clustering of ant larvae sorting and dead body clustering. Here a simple agent based model is designed and implemented inspired by the concept of stigmergy. The model uses agents moving on a two dimensional grid to reorganize clusters of larvae. This is called a model of self-organization. Emergent patterns seem eerily similar to actual ant colony brooding area dynamics.

Sometimes other ants are recruited to help when a single ant or small group of ants cannot finish a job. At other times agents may change their positions and alignments when trying to move an object. Swarm intelligence can simulate such teamwork. Each individual has its own simple set of instructions (for example, find the object, get next to it, and push it towards the goal). At first agents move randomly to try and accomplish the goal, but when they discover that they are moving away from their goal, some may change their position while others may try to move the object in a different direction. By continually applying such adjustments, these simple agents with limited individual capabilities can achieve the colony goal. Obviously, efficient ways of performing these tasks can be achieved by using a higher level of pre-codified human supplied intelligence and/or more communication among the more sophisticated agents. These simple agents with their meager abilities are able to accomplish the same goal through the power of self-organization and thus lend themselves to adaptation in many computer science applications.

Cemetery organization and brood (larvae) sorting are two important types of ant activities and are models for self-organization and data clustering. In some ant species the workers pile-up their colony's dead in order to clean their nests. Similarly, workers also sort the colony's brood in a systematic way. Small eggs are typically placed at the center of the nest, large larvae at the edge, and others in between in order of increasing size. This activity helps to control the temperature and care needs for eggs in various stages of development. Scientists have conjectured a simple explanation for this behavior. They hypothesize that ants pick-up, move, and put-down items guided by similarity to other objects in the surrounding area. These paradigms lend themselves to data classification/characterization (and graph partitioning problems). Since each record of data can be considered to be a point in a N-dimensional attribute space, a distance measurement between two records can be computed using a metric. In a data classification example, banks may use customer data they have on hand and group it to see which customers are more likely to pay back loans. If each record in a database is considered as a point with the distance between points calculated as described above the problem reduces to a data classification/characterization problem where similar points (i.e., at close distance) are grouped together on the grid (computer screen).

In this algorithm the data records are randomly are placed on the screen. Next a number of ants are randomly placed on the plane (grid). The ants are allowed to randomly walk around the grid, pick-up, move, and/or put-down nodes (data records). The ants pick-up and put-down a node with a probability, which is determined by the number of nodes this node and its neighboring nodes on the grid are close to as well as the number of uncommon nodes they are not close to. In essence the probability for pick-up is high when the node is near other nodes on the grid that it is not close to. The probability for putting-down a node is high when the node is near other nodes that it is close to. Using this simple algorithm the data file is partitioned into natural clusters.

The swarm model is not very efficient and may take many iterations before the data in the file is classified. This model has visual appeal since we can observe the data records being moved around and getting organized into clusters on the grid (computer screen).

<u>Swarm algorithm used for data classification.</u>

For every vertex $v_i$ do
    Place $v_i$ randomly on grid.
End For
For all agents do
    Place agent $A_m$ at randomly selected site.
End For
For t=1 to $t_{max}$ do
    For all agents do
        If ((agent unladen) and (site occupied by vertex $v_i$)) then
            Compute $f(v_i)$ and $p_p(v_i)$ = prob of pick up

Draw a random real number R between 0 and 1
If (R ≤ $p_p(v_i)$) then Pick up item $v_i$
End If
Else If ((agent carrying item $v_i$) and (site empty)) then
Compute $f(v_i)$ and $p_d(v_i)$ = prob of put down
Draw a random real number R between 0 and 1
If (R ≤ $p_d(v_i)$) then Drop item item $v_i$
End If
End If
Move the agent to a randomly selected neighboring site not occupied by another agent.
End For
Print location of vertices.
End For

$p_p(v_i) = (k_1/(k_1+f(v_i)))^2$, $t_{max}$ = # of iterations, $k_1$, $k_2$, and $\alpha$ are tuning factors

$p_d(v_i) = (f(v_i)/(k_2+f(v_i)))^2$, $f(v_i)$ =measure of similarity in an s by s neighborhood

$f(v_i) = 1/s^2 \Sigma_{vj\in \mathrm{Neigh}(s\times s)} [1- (d(v_i,v_j)/\alpha)]$ if $f(v_i) > 0$ and $f(v_i) = 0$ otherwise

$d(v_i, v_j) = (\Sigma_{k=1..n} |a_{ik}-a_{jk}|)/(\Sigma_{k=1..n} |a_{ik}| + \Sigma_{k=1..n} |a_{jk}|)$ = distance between data

## Self-Organizing Maps (an unsupervised neural network)

In computer science a neural network is a mathematical model that mimics and abstracts the functioning of networks of nerves in humans and animals. Specifically, the **self-organizing map** model attempts to simulate unsupervised learning (that is, the network organizes itself) in classifying the data in a data file [4, 5]. Initially the neurons $R_1$, $R_2$, …, $R_i$ ,…, $R_n$ , are assigned random unit vectors in real N-space and the data in the file is normalized to unit vectors. As new data is randomly presented, the closest representative neurons (forming a neighborhood) to that data are adjusted (using learning rate c) so that they will be more likely to be activated by similar data in the future. As more data is randomly presented, perhaps with repetition, the neighborhood sizes decrease and the learning rate decays exponentially thus creating neurons that recognize a certain class (cluster, category) of data from the data file. After training with historical data is complete, the neural net can now classify novel data records. This is accomplished by presenting the novel data to the self-organizing map and observing which neuron is activated most (i.e., which is closest in N-space. This neural representative specifies the class that this novel piece of data is in (that is, associated with). Since this method is unsupervised, the classes are somewhat unpredictable. This method is algorithmically efficient. However, since calculations and neurons are in real N-space, it is hard to visualize the actual clustering in progress.

Self-organizing Map Algorithm used for data classification.

For every data record X in the data file do
Normalize X to a unit vector in real N-space.
End For
For every representative $R_i$ do
Set $R_i$ to a random unit vector in real N-space.
End For
Repeat
Get X a random unit vector from the normalized data file.
For every representative $R_i$ do
Calculate dist($R_i$, X) (i.e., the Euclidean distance in N-space).
End For
Select $R_k$ for which the distance in minimum.
For every representative $R_i$ in the neighborhood of $R_k$ do
Set $R_i = R_i + c (X - R_i)$ (c is the learning rate)

and renormalize.
End For
Adjust the size of a neighborhood and exponentially lower c.
Until a terminal condition is reached.
Print location of vertices.
End Repeat

## A Hybrid Model

The hybrid model introduced here has the visual appeal of swarm intelligence and the efficiency of a self-organizing map.

Hybrid algorithm used for data classification.

```
For every data record X_i in the data file do
        Normalize X_i to a unit vector in real N-space.
        Assign X_i to v_I and place v_i randomly on grid.
End For
For every representative R_i do
        Set R_i to a random unit vector in real N-space.
        Place agent R_i at a randomly selected grid site so that no two are close.
End For
For all agents do
        Place agent A_m at a randomly selected site
End For
For t=1 to t_max do
        For all agents do
                If ((agent unladen) and (site occupied by vertex v_i)) then
                        Compute f(v_i) and p_p(v_i)
                        Draw a random real number R between 0 and 1
                        If (R ≤ p_p(v_i)) then Pick up item v_i
                        End If
                Else If ((agent carrying item v_i) and (site empty)) then
                        Compute f(v_i) and p_d(v_i)
                        Draw a random real number R between 0 and 1
                        If (R ≤ p_d(v_i)) then Drop item item v_i
                        End If
                End If
                If (agent carrying item v_i) then
                For every representative R_i do
                      Calculate dist(R_i, X) (Euclidean distance in N-space).
                End For
                Select R_k for which the distance in minimum.
                For every representative R_i in the N-space neighborhood
                of  R_k do
                      Set R_i = R_i + c(X – R_i)  (c =learning rate) & renormalize
                End For
                Adjust the size of a neighborhood and exponentially lower c.
                Move the agent in the direction of the closest R_i to a randomly
                selected neighboring site not occupied by another agent.
                Else Move the agent to a randomly selected neighboring site
                     not occupied by another agent
                End If
        End For
        Print location of vertices.
End For
```

As before, $p_p(v_i) = (k_1/(k_1+f(v_i)))^2$ and $p_d(v_i) = (f(v_i)/(k_2+f(v_i)))^2$ and

$f(v_i) = 1/s^2 \Sigma_{vj\in \text{Neigh}(s\times s)} [1- (d(v_i,v_j)/\alpha)]$ if $f(v_i) > 0$ and $f(v_i) = 0$ otherwise

Here, $d(v_i, v_j)$ = Euclidean distance between data records.

## Conclusion

Ant colonies and self-organizing maps have become two important and powerful classification heuristics in computer science and artificial intelligence. The novel hybrid model introduced here has the visual appeal of swarm intelligence and the efficiency of a self-organizing map.

## Acknowledgements

We wish to thank Pace University's School of Computer Science and Information Systems for partially supporting this research. We also thank DIMACS for giving Gargano and Lurie a unique reconnect research opportunity in summer, 2003 where we first discussed and investigated this hybrid model.

## References

[1] Bonabeau, Eric, et al, Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, 1999.

[2] Bonabeau, Eric, and Theraulaz, Guy, Swarm Smarts. Scientific American, Mar. 2000, p. 72-29.

[3] Gargano,M.R., Ant Colonies., MS/CS Masters Thesis, Pace University, 2001.

[4] Obermayer, Klaus and Sejnowski, Terrance J., Self-Organizing Map Formation. MIT Press, 2001.

[5] Wasserman, Phillip D., Neural Computing: Theory and Practice. Van Nostrand Reinhold, 1989.