Swarm intelligence

Yichen Hu

Abstract

Swarm intelligence is an important concept in artificial intelligence and computer science with emergent properties. The essential idea of swarm intelligence algorithms is to employ many simple agents applying almost no rule which in turn leads to an emergent global behavior. In this paper, we will introduce some of the most famous bio-mimicry algorithms and discuss their applications as well as metaheuristics inspired from the collective behaviors of nature.

1 Introduction

To put it in a simple way, swarm intelligence can be described as the collective behavior emerged from social insects working under very few rules. Self-organization is the main theme with limited restrictions from interactions among agents. Many famous examples of swarm intelligence come from the world of animals, such as birds flock, fish school and bugs swarm. The social interactions among individual agent help them to adapt to the environment more efficiently since more information are gathered from the whole swarm. This paper aims to introduce several well-known and interesting algorithms based on metaheuristic derived from nature and their applications in problem solving.

1.1 General principles

To model the broad behaviors arisen from a swarm, we introduce several general principles for swarm intelligence[1]:

Proximity principle The basic units of a swarm should be capable of simple computation related to its surrounding environment. Here computation is regarded as a direct behavioral response to environmental variance, such as those triggered by interactions among agents. Depending on the complexity of agents involved, responses may vary greatly. However, some fundamental behaviors are shared, such as living-resource searching and nest building.

Quality principle Apart from basic computation ability, a swarm should be able to response to quality factors, such as food and safety.

Principle of diverse response Resources should not be concentrated in narrow region. The distribution should be designed so that each agent will be maximally protected facing environmental fluctuations.

Principle of stability and adaptability Swarms are expected to adapt environmental fluctuations without rapidly changing modes since mode changing costs energy.

2 Ant colony optimization algorithms

The most recognized example of swarm intelligence in real world is the ants. To search for food, ants will start out from their colony and move randomly in all directions. Once a ant find food, it returns to colony and leave a trail of chemical substances called pheromone along the path. Other ants can then detect pheromone and follow the same path. The interesting point is that how often is the path visit by ants is determined by the concentration of pheromone along the path[3]. Since pheromone will naturally evaporate over time, the length of the path is also a factor. Therefore, under all these considerations, a shorter path will be favored because ants following that path keep adding pheromone which makes the concentration strong enough to against evaporation. As a result, the shortest path from colony to foods emerges.

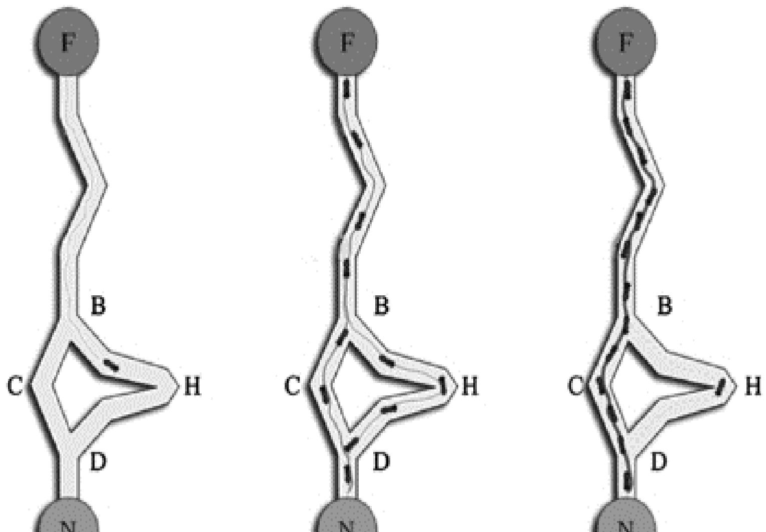

Figure 1: Ants select the shortest path

Learning from behaviors of real ants, a metaheuristic has been formulated for solving combinatorial problems, such as travelling salesman problem.

Define a combinatorial problem P as a triplet (S, Ω, f)[3]:

1. S is the search space over discrete decision variables $X_i \in D_i = \{v_i^1, \dots, v_i^{|D_i|}\}$.
2. Ω denotes the set of constraints.
3. Objective function $f : S \to \mathbb{R}$ to be maximized or minimized.

a solution $s \in S$ assigns values to variables that satisfies Ω and it asks for a solution $s^* \in S$ such that $f(s^*)$ is the global minimum or maximum.

The way that ant colony optimization algorithms tackle problems in this category is to employ the concept of pheromone. The ACO metaheuristic is split into three phases[2]:

```
Initialization;
while not terminated do
    Construct solution using artifical ants;
    Local search (optional);
    Update pheromones;
end
```

Algorithm 1: The ACO metaheuristic

Solution construction Using m artificial ants, solutions $C = \{c_{ij}\}$, $i = 1 \dots n$, $j = i \dots D_i$ satisfies all the constraints Ω are constructed, where c_{ij} assigns the decision variable $X_i = v_i^j$. This can be also viewed as a random walk of ants on the construction graph $G_c(V, E)$.

Local search With specific design for individual problem, a local search could improve the constructed solution. However, since this is highly variable according to problems, local search is an optional process.

Update pheromones Pheromone values for promising solutions will be increased and values for undesired solutions will be decreased by pheromone evaporation. Thus the best solutions will be rewarded with the highest concentration of pheromones.

Many NP-hard problems in computer science, which are problems with exponential time worst case complexity, can be solved using ACO algorithms, such as the assignment problem category and the scheduling problem category[2]. There are proofs that ACO algorithms will converge to these best-performing algorithms. However, the speed of convergence is unknown and the performance of ACO algorithms largely depend on if an optimal local search procedure can be found and this is very problem-specific.[3]

3 Bee colony optimization algorithms

Just like ants, bees have similar food collecting behaviors. Instead of pheromones, bees colony optimization algorithm relies on the foraging behavior of honey bees. At the first stage, some bees are sent out to look for promising food sources. After a good food source is located, bees return back to colony and perform a waggle dance to spread out information about the source. Three pieces of information are included:1. distance, 2. direction, 3. quality of food source. The better the quality of food source, the more bees will be attracted. Therefore, the best food source emerges[4].

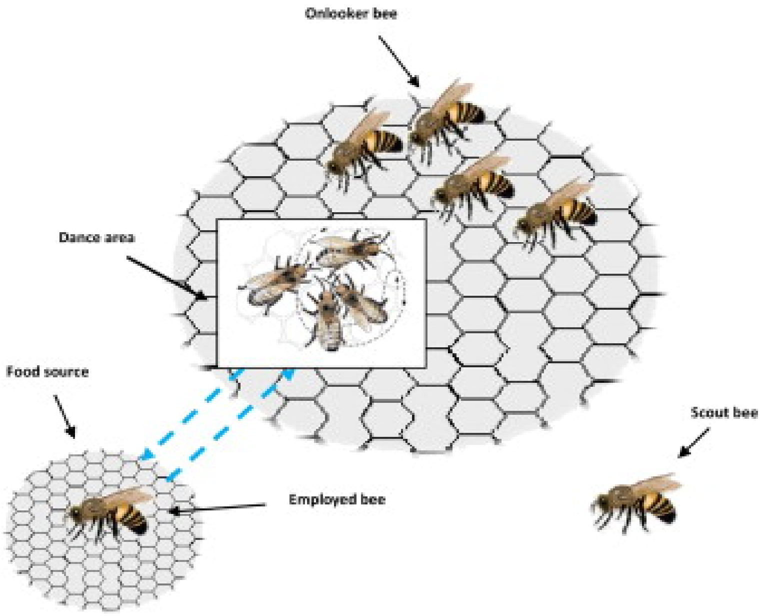

Figure 2: How bees work to find food sources

The metaheuristic extracted from the foraging behaviors of bees can also be applied to solve combinatorial problems; especially problems involve global minimum or maximum. Similarly, the BCO metaheuristic undergo several phases[5]:

Initialization;
while *not terminated* **do**
    Employed Bees Phase;
    Onlooker Bees Phase;
    Scout Bees Phase;
    Memorize the best solution;
end

Algorithm 2: The BCO metaheuristic

Initialization All the food sources $\vec{F_m}$, $m = 1, \ldots, N$ are initialized. $\vec{F_m}$ are solutions to the optimization problems and will be tuned by BCO algorithm to minimize or maximize objective function f defined above.

Employed Bees Employed bees will search in the neighborhood of $\vec{F_m}$ from memory with a random vector $\vec{R_m}$. A fitness will be calculated to determine if $\vec{R_m}$ leads to a better food source. The usual choice for fitness function T is:

$$T(\vec{x_m}) = \begin{cases} \frac{1}{1+f(\vec{x_m})}, & \text{if } f(\vec{x_m}) \geq 0 \\ 1 + |f(\vec{x_m})|, & \text{if } f(\vec{x_m}) < 0 \end{cases} \tag{1}$$

Onlooker Bees After employed bees shared information about food sources, onlooker bees will probabilistically choose their destination accordingly. Usually, this is calculated depending on the fitness values provided by employed bees. For example, with the above defined fitness value $T(\vec{x_m})$, the probability value p_m can be calculated:

$$p_m = \frac{T(\vec{x_m})}{\sum_{m=1}^{N} T(\vec{x_m})} \tag{2}$$

With more onlooker bees recruited to richer resources, positive feedback also arises for richer resources.

Scout bees The third kind of bees is the scout bees. They are usually these employed bees abandoned by the algorithms because the quality of food sources they found is poor. Scout bees will again start from the beginning and search for food sources randomly. However, negative feedback will lower the attractiveness of their previous found food sources.

The BCO algorithms have interesting applications in numerical optimizations, for example, it can be used to find global optimal solutions of functions. Moreover, recent studies suggest that the BCO algorithms can also be applied to problems in shop scheduling[8], neural network training[6] and imaging processing[7].

4 Particle swarm optimization algorithms

Bird flocking and fish schooling are the inspirations from nature behind particle swarm optimization algorithms. It was first proposed by Eberhart and Kennedy[9]. Mimicking physical quantities such as velocity and position in bird flocking, artificial particles are constructed to “fly” inside the search space of optimization problems.

Figure 3: Bird flocks

However, different from the previous two algorithms using pheromone or feedback as tools to get rid of undesired solutions, particle swarm optimization algorithms updates the current solution directly. As you can tell from the following description of the framework of PSO algorithms, with fewer parameters, PSO algorithms are easy to implement and achieve global optimal solutions with high probability.

Initially, a population of particles is distributed uniformly in the multi-dimension search space of the objective function of the optimization problem. Two quantities are associated with particles, a position vector $\vec{x_i}$ and a velocity $\vec{v_i}$. At each time step, the velocities of particles will be updated according to the following formula[9]:

$$\vec{v_i}^{t+1} = v_i^t + r_1 * \alpha(\vec{b} - \vec{x_i^t}) + r_2 * \beta(\vec{n} - \vec{x_i^t}) \tag{3}$$

where $\vec{b}$ is the global best location and $\vec{n}$ is the best location in the neighborhood of particle p_i. Both α,β are learning parameters and r_1, r_2 are random parameters ranging from 0 to 1.

The positions will be updated simply by[9]

$$\vec{x_i^{t+1}} = \vec{x_i^t} + \vec{v_i^{t+1}} \tag{4}$$

The importance of including a neighborhood best location $\vec{n}$ for each particle is to avoid the swarm being trapped into a local minimum. This is where the social connection comes into play in particle swarm optimizations. The social connection topology is called swarm's population topology. There are different types, for example, the gbest topology. With such a topology, particles are all attracted to the global best solution; therefore, it represents the fully connected social network. The lbest topology, on the other hand, connects each particle with only C neighbors. This kind of topology slows the process of convergence compared to gbest, however, it also makes the PSO algorithms more capable of avoiding local minima. The wheel topology connects neighboring particles to only one particle–the focal particle. This type is the one with the smallest number of connections[10].

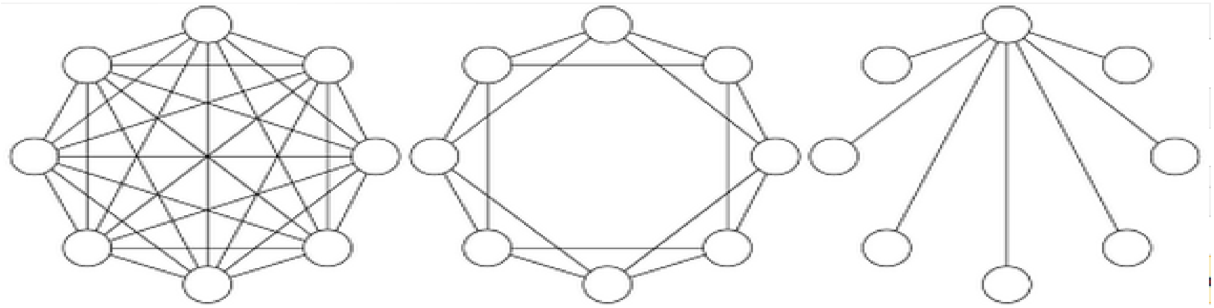

Figure 4: From left to right: gbest, lbest, wheel topology

5 Cuckoo search

Cuckoo search was invented by Xin-she Yang and Suash Deb in 2009[11]. This algorithm is inspired by the brood parasitism behavior of some species of cuckoo. They will lay their eggs in other bird's nest. If the host bird find out about this, it will either throw away the intruding egg or simply abandon the whole nest and start a new one. However, some species of cuckoo are very good at making their eggs the same as the host's egg, and therefore greatly increase the survival probability of their eggs.

Basic rules of cuckoo search[11]:

- Each cuckoo lays one egg at a time and dumps it in a randomly chosen nest.
- The best nests with high quality of eggs will be brought to the next generation.
- The number of host nests is fixed. A host bird will discover the egg is laid by cuckoo by a probability $p_\alpha \in (0,1)$. The host bird can get rid of the egg or build a new nest.

Any egg x_i in the nest represent a solution to the problem, and at each time step, we use the following to generate new solutions[11]:

(5) $$\vec{x_i^{t+1}} = \vec{x_i^t} + \alpha \oplus Levy(\lambda)$$

where α is the step size related to scales of the problem and λ is the parameter of Levy flight$= u^{-\lambda}$.

Initialization of hosts;
while *not terminated* **do**

   Let a cuckoo generate a new solution using the above Levy flight process and calculate the fitness F_i;
   Choose a nest j randomly;
   If $(F_i < F_j)$ then replace solution j ;
   Abandon p_α percent of the nests with new ones built;
   Memorize the best solutions

end

Algorithm 3: Cuckoo search[15]

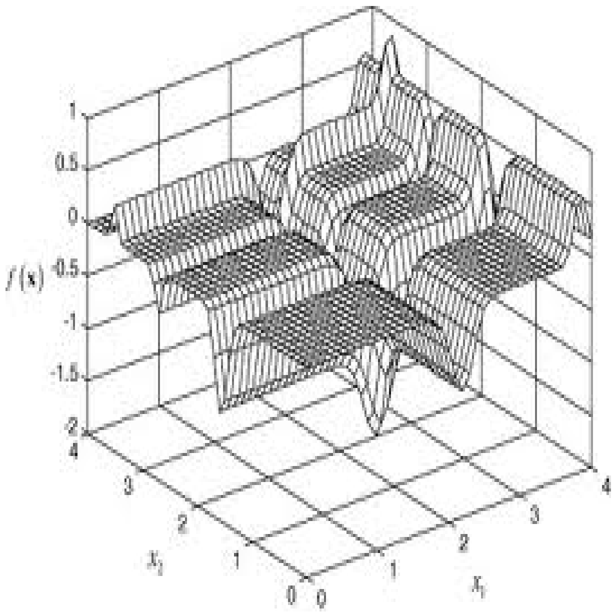

Figure 5: 2D Michalewicz' function with many minimums

Let us apply cuckoo search on a test function-the 2D Michalewicz' function, to find global minimum with 20 nests and p_α=0.25.

$$f(\vec{x}) = -\sum_{i=1}^{2} sin(x_i)[sin(\frac{ix_i^2}{\pi})]^{2m} \quad (6)$$

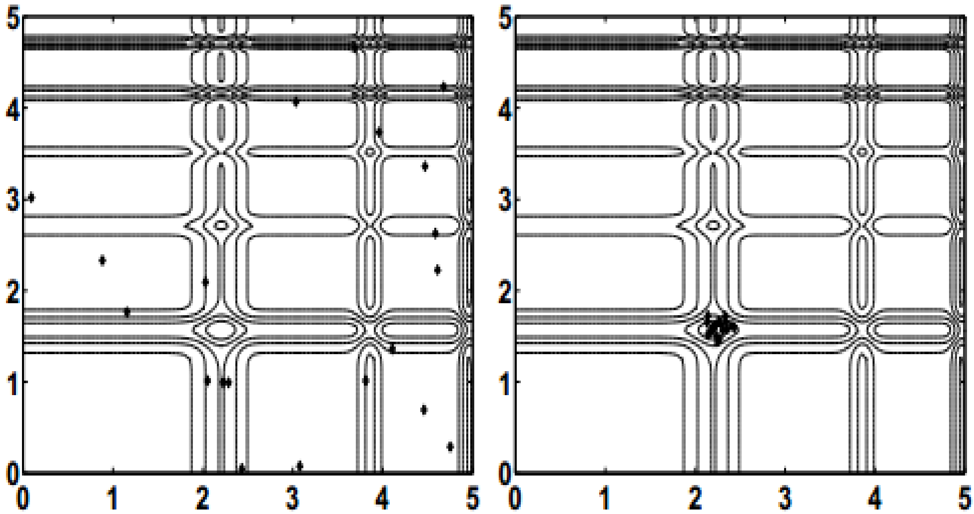

Figure 6: From left to right: Initial and final positions of the nests marked using dots[12]

Compared with the particle swarm optimization algorithms introduced above, cuckoo search has several advantages and outperform the PSO algorithms[12]. First of all is its usage of Levy flight for random walk. Levy flight shares certain common grounds with the flight behaviors of animals and insects in nature[14]. Therefore, instead of having uniform or Gaussian distributed random generator, Levy flight produces more suitable and efficient step size for nests in search space. Also, unlike particle swarm, cuckoo search has less parameter that needs to be fine-tuned.

6 Conclusion

In this paper, I introduced four interesting nature inspired algorithms in swarm intelligence. The ant and bee colony optimization algorithms are based on metaheurisitic derived from their feedback systems: pheromones and waggle dances. The information accumulated from their collective behaviors guides each agent toward the optimal solution. Particle swarm optimization algorithms have the simplest framework, employing particles to search for optimal solution directly. With the help of social interactions and related population topologies, PSO algorithms are able to avoid local minimums and search for global optimal solution more efficiently. Cuckoo search is one of the most interesting algorithms. It mimics the brood parasitism of cuckoos. By protecting better solutions and abandon undesired ones throughout evolution, the global optimal solution emerges. All of the four algorithms have rich applications in problem solving. They can be customized to deal with problems such as NP-hard problems, combinatorial optimization problems and numerical optimization problems. On the other hand, we should also note the limitations of swarm intelligence. First of all, all swarm intelligence algorithms have parts that are problem-specific, for example, the performance of ant colony optimization algorithms largely depends on the local search subroutines and population topologies have influences on the performance of particle swarm optimization algorithms. Secondly, it is hard to analyze the computational complexity of these algorithms[16, 3], therefore, it is difficult to tell whether a swarm intelligence algorithm will be suitable or efficient for certain problems. Moreover, the famous "no free lunch theorem" by Wolpert and Macready[17] questions the overall performance of swarm intelligence algorithms and argues that if we take an average over all problems, every optimization algorithm works equally well.

References

[1] M. Millonas. *Swarms, Phase Transitions, and Collective Intelligence.* Addison-Wesley Publishing Company, Reading (1994).

[2] M. Dorigo, and T. Sttzle. *Ant Colony Optimization.* A Bradford Book, (2004).

[3] M. Dorigo, T. Sttzle, and M. Birattari. (2006). Ant colony optimization. *Computational Intelligence Magazine*, 1(4), 28-39.

[4] C. P. Lim, L. C. Jain, and S. Dehuri. *Innovations in Swarm Intelligence*, Springer, (2009).

[5] X.-S. Yang. *Introduction To Computational Mathematics*, World Scientific Publishing Company, (2008).

[6] D. Karaboga, and C. Ozturk. (2009). Neural Networks Training by Artificial Bee Colony Algorithm on Pattern Classification. *Neural Network World*, 19 (3), 279-292.

[7] C. Xu, and H. Duan. (2010). Artificial bee colony (ABC) optimized edge potential function (EPF) approach to target recognition for low-altitude aircraft. *Pattern Recognition Letters*, Elsevier, Netherlands, In Press.

[8] N. Quijano, and K. Passino. (2007). Honey bee social foraging algorithms for resource allocation *American Control Conference*, ACC '07, pages 3383-3388, New York.

[9] J. Kennedy, and R. C. Eberhart. (1995). Particle swarm optimization. *In Proceedings of the 1995 IEEE International Conference on Neural Networks*, Vol. 4, pp. 1942-1948

[10] D. A. Pelta, N. Krasnogo, D. Dumitrescu, C. Chira, and R. Lung. *Nature Inspired Cooperative Strategies for Optimization*, Springer, (2011).

[11] X.-S. Yang, S. Deb. (2009). Cuckoo search via Levy ights. *Proc. of World Congress on Nature Biologically Inspired Computing*, pp. 210-214.

[12] X.-S. Yang, S. Deb. (2010). Engineering Optimisation by Cuckoo Search. *Int. J. Mathematical Modelling and Numerical Optimisation*, 1(4), 330-343.

[13] X.-S. Yang. *Nature-Inspired Metaheuristic Algorithms: Second Edition*, Luniver Press, (2010).

[14] C. Brown, L. S. Liebovitch, R. Glendon. (2007). Levy ights in Dobe Ju/hoansi foraging patterns. *Human Ecology*, 35, 129-138.

[15] X.-S. Yang. *Nature-Inspired Metaheuristic Algorithms: Second Edition*, Luniver Press, (2010).

[16] I. C. Trelea. (2003). The Particle Swarm Optimization Algorithm: convergence analysis and parameter selection. *Information Processing Letters*, 85 (6), 317325.

[17] D. H. Wolpert, W. G. Macready. (1997). No Free Lunch Theorems for Optimization. *IEEE Transactions on Evolutionary Computation*, 1, 67.