Swarm Intelligence

Simran Bhatti, Vandana Tayal, Pooja Gulia
Student, Computer Science, Maharishi Dayanand University
Gurgaon, Haryana, India

Abstract- Swarm Intelligence (SI) is a computational intelligence technique involving the study of collective behavior in decentralized systems. Such systems are made up by a population of simple individuals interacting locally with one other and with their environment. Although there is typically no centralized control dictating the behavior of the individuals, local interactions among individuals often cause a global pattern to emerge. Examples of systems like this can be found in nature, including ant colonies, bird flocking, animal herding, honey bees, and many more.

***Index Terms*- computational, decentralized, intelligence, dictating, centralized, ant colonies.**

I. INTRODUCTION

Despite the current technology, and rapid advances in every field, there are still some problems that continue to elude scientists. Complex NP complete problems, vehicle routing, network maintenance, the traveling salesperson, and computing the shortest route between two points are just a couple of examples of these types of problems. Learning algorithms have been developed in conjunction with artificial intelligence systems such as neural networks to try and solve some of these problems, but imperfections and inefficiencies in both the hardware and software have prevent reliable results.

Genetic algorithms also made an attempt at these problems, and had some success. The algorithms were considered too complex to re-implement, realizing the random nature of the mutation. Genetic algorithms were also not reliable enough to be considered the best source for solving these complex problems. Scientists, now, are looking into the world of insects in search of new methods and approaches of attacking complex problems.

II. ALGORITHMS

Ant colony optimization

Ant colony optimization or ACO is an optimization algorithm that can be used to find approximate solutions to difficult combinatorial optimization problems. In ACO artificial ants build solutions by moving on the problem graph and they, mimicking real ants, deposit artificial pheromone on the graph in such a way that future artificial ants can build better solutions. ACO has been successfully applied to an impressive number of optimization problems.

Particle swarm optimization

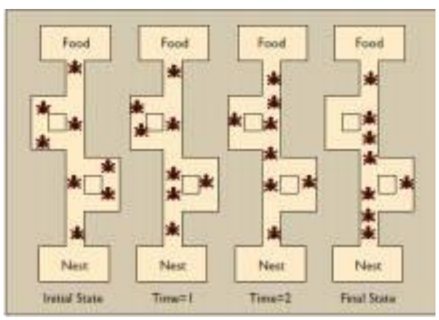
Particle swarm optimization or PSO is a global optimization algorithm for dealing with problems in which a best solution can be represented as a point or surface in an n-dimensional space. Hypotheses are plotted in this space and seeded with an initial velocity, as well as a communication channel between the particles. Particles then move through the solution space, and are evaluated according to some fitness criterion after each time step. Over time, particles are accelerated towards those particles within their communication grouping which have better fitness values.

Stochastic diffusion search

Stochastic Diffusion Search or SDS is an agent based probabilistic global search and optimization technique best suited to problems where the objective function can be decomposed into multiple independent partial-functions. Each agent maintains a hypothesis which is iteratively tested by evaluating a randomly selected partial objective function parameterized by the agent's current hypothesis. In the standard version of SDS such partial function evaluations are binary resulting in each agent becoming active or inactive. Information on hypotheses is diffused across the population via inter-agent communication. Unlike the stigmergic communication used in ACO, in SDS agents communicate hypotheses via a one-to-one communication strategy analogous to the tandem running procedure observed in some species of ant. A positive feedback mechanism ensures that, over time, a population of agents stabilizes around the global-best solution. SDS is both an efficient and robust search and optimization algorithm, which has been extensively mathematically described.

The Ant Behavior

The ants must leave their nest in search of food, but the colony doesn't know where any food is currently located. The individual ants make their decisions on which direction to go just on chance, they have just as much probability of going one direction as another. As soon as an ant finds food it will take some back to the nest. As ants move they leave behind a chemical substance called pheromone, which other ants can smell and identify that an ant has been there before.



Ants converging on the shortest path between a food source and the nest

The ants that find the closest food source will be able to leave more amounts of pheromone since their travel time is less then the other ants. When other ants finally return to the nest, they will decide where to go based upon the amount of pheromone that they smell. "The stronger the pheromone level, the more likely an ant is to take that route." As more and more ants take the shorter path, the pheromone level increases until every ant is taking the same shorter path.

Ant Colony Optimization Algorithm

The natural behavior of these ants can be programmed into an *ant algorithm*, which we can use to find the shortest path within graphs. Since most problems can be reduced to a graph, being able to find the shortest path through a graph often provides the simplest and fastest solution to the problem. Each ant can be represented by an agent in the program that has only has limited abilities, such as sensing and dispensing pheromone, and a memory so that any agent can trace its steps backwards through the graph in order to determine the length of the tour. When an agent senses another agent's pheromone it will receive positive feedback in deciding to choose the popular path. The positive feedback triggers *autocatalysis*, meaning that a small amount of positive feedback results in more positive feedback and so on and so forth.

There are a couple of conditions that must be met in order for the agents to successfully navigate any kind of graph. The pheromone must have an evaporation period because it allows for the agents to converge in the first place. If the pheromone did not evaporate, then agents will always be choosing between multiple paths that seem equally as popular. As the agents converge on a path, they must become less and less influenced by other paths, especially if there are no longer any agents that are traversing them. There is a second subtle problem of the agents converging on a sub-optimal path before the shortest path is discovered. In the example of the ants, it is possible that there is a food source closer to the nest, but it had not been found before the colony converged on a different food source. This problem is remedied by the fact that the agent's/ant's decision is not fixed; it always has a chance of not choosing the popular route, and thus exploring for alternate routes. The construction of alternate paths is of great importance when it comes to using *ant algorithms* for problems other than shortest route.

Ant colony optimization algorithms have been used to produce near-optimal solutions to the traveling salesman problem. They have an advantage over simulated annealing and genetic algorithm approaches when the graph may change dynamically; the ant colony algorithm can be run continuously and adapt to changes in real time. This is of interest in network routing and urban transportation systems.The use of agents produces faster and more optimal solutions to problems.

Problem	Genetic Algorithms	Simulated Annealing	Tabu Search	Neural Networks
Traveling Salesman	Better	Better	Same	---
Vehicle Routing	---*	Better	Worse	Better
Single Machine Tardiness	---	Better	---	---
Power Economic Dispatch	Better	---	---	---

*Not tested Table summarized from data in [2, 4, 5, 13]

Overall performance of ant approach compared to other techniques for several problems.

Particle Swarm Optimization Algorithm

Particle swarm optimization (PSO) is a form of swarm intelligence. Imagine a swarm of insects or a school of fish. If one sees a desirable path to go (e.g., for food, protection, etc.) the rest of the swarm will be able to follow quickly even if they are on the opposite side of the swarm. On the other hand, in order to facilitate felicitous exploration of the search space, typically one wants each particle to have a certain level of "craziness" or randomness in their movement, so that the movement of the swarm has a certain explorative capability: the swarm should be influenced by the rest of the swarm but also should independently explore to a certain extent. (This is a manifestation of the basic exploration-exploitation tradeoff that occurs in any search problem.)

This is modeled by particles in multidimensional space that have a position and a velocity. These particles are flying through hyperspace (i.e., $\mathbb{R}^n$) and have two essential reasoning capabilities: their memory of their own best position and knowledge of the swarm's best, "best" simply meaning the position with the smallest objective value. Members of a swarm communicate good positions to each other and adjust their own position and velocity based on these good positions. There are two main ways this communication is done:

- a global best that is known to all and immediately updated when a new best position is found by any particle in the swarm
- "neighborhood" bests where each particle only immediately communicates with a subset of the swarm about best positions

An algorithm is presented below where there is a global best rather than neighborhood bests. Neighborhood bests allow better exploration of the search space and reduce the susceptibility of PSO to falling into local minima, but slow down convergence speed. Note that neighborhoods merely slow down the proliferation of new bests, rather than creating isolated sub-swarms because of the overlapping of neighborhoods: to make neighborhoods of size 3, say, particle 1 would only communicate with particles 2 through 5, particle 2 with 3 through 6, and so on. But then a new best position discovered by particle 2's neighborhood would be communicated to particle 1's neighborhood at the next iteration of the PSO algorithm presented below. Smaller neighborhoods lead to slower convergence, while larger neighborhoods to faster convergence, with a global best representing a neighborhood consisting of the entire swarm.

III. PSEUDO CODE

Here follows a pseudo code example of the position update of the swarm. Note that the random vectors $\mathbf{r}_1, \mathbf{r}_2$ are implemented as scalars inside the dimension loop which is equivalent to the mathematical description given.

```
for I = 1 to number of particles n do
  for J=1 to number of dimensions m do
    R1=uniform random number
    R2=uniform random number
    V[I][J]=w*V[I][J]
           +C1*R1*(P[I][J]-X[I][J])
           +C2*R2*(G[I][J]-X[I][J])
    X[I][J] = X[I][J]+V[I][J]
  enddo
enddo
```

IV. DISCUSSION

By studying this algorithm, we see that we are essentially carrying out something like a discrete-time simulation where each iteration of it represents a "tic" of time. The particles "communicate" information they find about each other by updating their velocities in terms of local and global bests; when a new best is found, the particles will change their positions accordingly so that the new information is "broadcast" to the swarm. The particles are always drawn back both to their own personal best positions and also to the best position of the entire swarm. They also have stochastic exploration capability via the use of the random multipliers r_1,r_2. The vector, floating-point nature of the algorithm suggests that high-performance implementations could be created that take advantage of modern hardware extensions pertaining to vectorization, such as Streaming SIMD Extensions and Altivec.

Typical convergence conditions include reaching a certain number of iterations, reaching a certain fitness value, and so on.

V. STOCHASTIC DIFFUSION SEARCH

Stochastic Diffusion Search (SDS), was first described in 1989 as a population-based, pattern-matching algorithm. It belongs to a family of Swarm Intelligence and naturally inspired search and optimization algorithms which includes Ant Colony Optimization, Particle Swarm Optimization and Genetic Algorithms. Unlike stigmergetic communication employed in Ant Colony Optimization, which is based on modification of the physical properties of a simulated environment, SDS uses a form of direct (one-to-one) communication between the agents similar to the tandem calling mechanism employed by one species of ants, *Leptothorax acervorum*.

In SDS agents perform cheap, partial evaluations of a hypothesis (a candidate solution to the search problem). They then share information about hypotheses (diffusion of information) through direct one-to-one communication. As a result of the diffusion mechanism, high-quality solutions can be identified from clusters of agents with the same hypothesis. The operation of SDS is most easily understood by means of a simple analogy - The Restaurant Game.

VI. THE RESTAURANT GAME

A group of delegates attends a long conference in an unfamiliar town. Each night they have to find somewhere to dine. There is a large choice of restaurants, each of which offers a large variety of meals. The problem the group faces is to find the best restaurant, that is the restaurant where the maximum number of delegates would enjoy dining. Even a parallel exhaustive search through the restaurant and meal combinations would take too long to accomplish. To solve the problem delegates decide to employ a Stochastic Diffusion Search.

Each delegate acts as an agent maintaining a hypothesis identifying the best restaurant in town. Each night each delegate tests his hypothesis by dining there and randomly selecting one of the meals on offer. The next morning at breakfast every delegate who did not enjoy his meal the previous night, asks one randomly selected colleague to share his dinner impressions. If the experience was good, he also adopts this restaurant as his choice. Otherwise he simply selects another restaurant at random from those listed in `Yellow Pages'. Using this strategy it is found that very rapidly significant number of delegates congregate around the 'best' restaurant in town.

VII. APPLICATIONS

Swarm intelligence offers researchers and scientists a tool for solving very difficult NP class problems.

Swarm intelligence’s ability to solve these problems leads to various practical real world applications such as, traffic routing, networking, games, industry, robotics, and perhaps even simulating the global economy.

Although the technology is relatively new, already scientists are realizing its potential. The use of ant algorithms within computing systems has helped to solidify swarm intelligence’s place in the computing world. Already researchers are observing other social animals, such as bees and schools of fish in order to discover how swarm behavior of those animals might be utilized in future applications and algorithms.

VIII. SWARM ROBOTS

Swarm robotics is currently one of the most important application areas for swarm intelligence. Swarms provide the possibility of enhanced task performance, high reliability (fault tolerance), low unit complexity and decreased cost over traditional robotic systems. They can accomplish some tasks that would be impossible for a single robot to achieve. Swarm robots can be applied to many fields, such as flexible manufacturing systems, spacecraft, inspection/maintenance, construction, agriculture, and medicine work.

Many different swarm models have been proposed. Beni introduced the concept of cellular robotics systems, which consists of collections of autonomous, non-synchronized, non-intelligent robots cooperating on a finite n-dimensional cellular space under distributed control. Limited communication exists only between adjacent robots. These robots operate autonomously and cooperate with others to accomplish predefined global tasks.

Swarm robots are more than just networks of independent agents, they are potentially reconfigurable networks of communicating agents capable of coordinated sensing and interaction with the environment

IX. SWARM TO COMBAT MARINE OIL POLLUTION

While using mechanical ants in sky-high buildings is still part of the distant future, a project of swarm robotics is already taking shape in the sea. Researchers at the Fraunhofer Institute for Manufacturing Engineering and Automation (IPA), Germany are gearing towards using this technology to control oil spill.

The robots in this case are small ships resembling catamarans with an exhaust device between the vats. It is not possible to have a central planning system since the prediction of the nature of an oil spill is unreliable. Thus, 100 robots after taking their respective positions, communicate with one another, find the oil and then absorb it.

As long as a sensor communicates its presence in the oil spill to the robot, the process is smooth. The robot rotates coils until it discovers the oil. The number of robots is directly proportional to how well the waste can be eliminated. Currently, this method is being used in a simulation with small mobile robots. A beamer at the base indicates an expanding oil spill.

X. OTHER APPLICATIONS

Swarm Intelligence-based techniques can be used in a number of applications.

The U.S. military is investigating swarm techniques for controlling unmanned vehicles.

ESA is thinking about orbital swarm for self assembly and interferometry.

NASA is investigating the use of swarm technology for planetary mapping.

A 1992 paper by M. Anthony Lewis and George A. Bekey discusses the possibility of using swarm intelligence to control nanobots within the body for the purpose of killing cancer tumors.

Artists are using swarm technology as a means of creating complex interactive environments. Disney's *The Lion King* was the first movie to make use of swarm technology (the stampede of the wild beasts scene). The movie "Lord of the Rings" has also made use of similar technology during battle scenes.

Swarm technology is particularly attractive because it is cheap, robust, and simple.