International Journal of Computer Science and Business Informatics

IJCSBI.ORG

Comparison of Swarm Intelligence Techniques

Prof. S. A. Thakare
Assistant Professor
JDIET, Yavatmal (MS), India.

ABSTRACT

Swarm intelligence is a computational intelligence technique to solve complex real-world problems. It involves the study of collective behaviour of behavior of decentralized, self-organized systems, natural artificial. Swarm Intelligence (SI) is an innovative distributed intelligent paradigm for solving optimization problems that originally took its inspiration from the biological examples by swarming, flocking and herding phenomena in vertebrates. In this paper, we have made extensive analysis of the most successful methods of optimization techniques inspired by Swarm Intelligence (SI): Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO). An analysis of these algorithms is carried out with fitness sharing, aiming to investigate whether this improves performance which can be implemented in the evolutionary algorithms.

1. INTRODUCTION

Swarm intelligence (SI) is "*The emergent collective intelligence of groups of simple agents*" (Bonabeau et al., 1999). It gives rise to complex and often intelligent behavior through simple, unsupervised (no centralized control) interactions between a sheer number of autonomous swarm members. This results in the emergence of very complex forms of social behavior which fulfills a number of optimization objectives and other tasks. Swarm is considered as biological insects like ants, bees, wasps, fish etc. In this paper we have considered biological insects' ant and birds flocking for our study. Ants possess the following characteristics:

(1) Scalability: The ants can change their group size by local and distributed agent interactions. This is an important characteristic by which the group is scaled to the desired level.

(2) Fault tolerance: Each ant follows a simple rule. They do not rely on a centralized control mechanism, graceful, scalable degradation.

(3) Adaptation: Ants always search for new path by roaming around their nest. Once they find the food their nest members follow the shortest path. While nest members follow shortest path, some of the members of the colony always search for another shortest path. To accomplish this they change, die or reproduce for the colony.

(4) Speed: In order to make other ants to know the food source, they move faster to their nest. Other ants find more pheromone on the path and follow the path to the food source. Thus changes are

propagated very fast to communicate to other nest mates in order to follow the food source.

(5) Modularity: Ants follow the simple rule of following the path which has a higher level of pheromone concentration. They do not interact directly and act independently to accomplish the task.

(6) Autonomy: No centralized control and hence no supervisor is needed. They work for the colony and always strive to search food source around their colony.

(7) Parallelism: Ants work independently and the task of searching food source is carried out by each ant in parallelism. It is parallelism due to which they change their path, if a new food source is found near their colony. These characteristics of biological insects such as ants resemble the characteristics of Mobile Ad Hoc Networks. This helps us to apply the food searching characteristics of ants for routing packets in Mobile Ad Hoc Networks.

2. SWARM INTELLIGENCE (SI) MODELS

Swarm intelligence models are referred to as computational models inspired by natural swarm systems. To date, several swarm intelligence models based on different natural swarm systems have been proposed in the literature, and successfully applied in many real-life applications. Examples of swarm intelligence models are: Ant Colony Optimization, Particle Swarm Optimization, Artificial Bee Colony, Bacterial Foraging, Cat Swarm Optimization, Artificial Immune System, and Glowworm Swarm Optimization. In this paper, we will primarily focus on two of the most popular swarm intelligence models, namely, Ant Colony Optimization and Particle Swarm Optimization.

2.1 Ant Colony Optimization (ACO) Model

The ant colony optimization meta-heuristic is a particular class of ant algorithms. Ant algorithms are multi-agent systems, which consist of agents with the behavior of individual ants [1]. The ant colony optimization algorithm (ACO) is a probabilistic technique for solving computational problems which can be reduced to finding better paths through graphs. In the real world, ants (initially) wander randomly, and upon finding food return to their colony while laying down pheromone trails. If other ants find such a path, they are likely not to keep travelling at random, but to instead follow the trail; returning and reinforcing it if they eventually find food.

Over time, however, the pheromone trail starts to evaporate, thus reducing its attractive strength. The more time it takes for an ant to travel down the path and back again, the more time the pheromones have to evaporate. A short path, by comparison, gets marched over faster, and thus the

pheromone density remains high as it is laid on the path as fast as it can evaporate. Pheromone evaporation has also the advantage of avoiding the convergence to a locally optimal solution. If there were no evaporation at all, the paths chosen by the first ant would tend to be excessively attractive to the following ones. In that case, the exploration of the solution space would be constrained [3]. Thus, when one ant finds a good (i.e., short) path from the colony to a food source, other ants are more likely to follow that path, and positive feedback eventually leads all the ants following a single path. The idea of the ant colony algorithm is to mimic this behavior with "simulated ants" walking around the graph representing the problem to solve.

The original idea comes from observing the exploitation of food resources among ants, in which ants' individually limited cognitive abilities have collectively been able to find the shortest path between a food source and the nest [4, 5, 7].

1. The first ant finds the food source (F), via any way (a), then returns to the nest (N), leaving behind a trail of pheromone (b)
2. Ants indiscriminately follow four possible ways, but the strengthening of the runway makes it more attractive as the shortest route.
3. Ants take the shortest route; long portions of other ways lose their trail pheromones.

In a series of experiments on a colony of ants with a choice between two unequal length paths leading to a source of food, biologists have observed that ants tended to use the shortest route. A model explaining this behavior is as follows:

1. An ant (called "blitz") runs more or less at random around the colony;
2. If it discovers a food source, it returns more or less directly to the nest, leaving in its path a trail of pheromone;
3. These pheromones are attractive, nearby ants will be inclined to follow, more or less directly, the track;
4. Returning to the colony, these ants will strengthen the route;
5. If two routes are possible to reach the same food source, the shorter one will be, in the same time, traveled by more ants than the long route will.
6. The short route will be increasingly enhanced, and therefore become more attractive;
7. The longer route will eventually disappear, pheromones are volatile;
8. Eventually, all the ants have determined and therefore "chosen" the shortest route. Ants use the environment as a medium of

communication. They exchange information indirectly by depositing pheromones, all detailing the status of their "work". The information exchanged has a local scope, only an ant located where the pheromones were left has a notion of them. This system is called "Stigmergy" and occurs in many social animal societies (it has been studied in the case of the construction of pillars in the nests of termites). The mechanism to solve a problem too complex to be addressed by single ants is a good example of a self-organized system. This system is based on positive feedback (the deposit of pheromone attracts other ants that will strengthen it by themselves) and negative (dissipation of the route by evaporation prevents the system from thrashing). Theoretically, if the quantity of pheromone remained the same over time on all edges, no route would be chosen. However, because of feedback, a slight variation on an edge will be amplified and thus allow the choice of an edge. The algorithm will move from an unstable state in which no edge is stronger than another, to a stable state where the route is composed of the strongest edges.

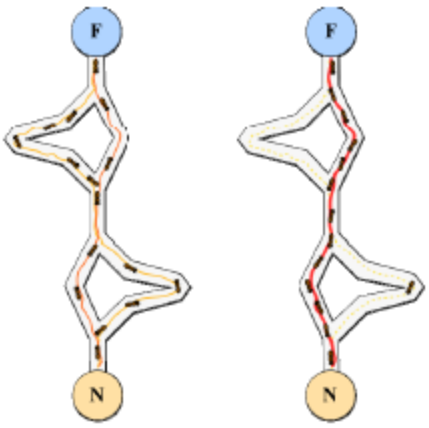

Figure 1. Behavior of real ant movements

2.1.1 Ant Colony Optimization meta-heuristic Algorithm

Let $G = (V, E)$ be a connected graph with $n = |V|$ nodes. The simple ant colony optimization meta-heuristic can be used to find the shortest path between a source node *vs* and a destination node *vd* on the graph *G* [7]. The path length is given by the number of nodes on the path. Each edge $e(i, j) \in E$ of the graph connecting the nodes *vi* and *vj* has a variable $\phi i,j$ (artificial pheromone), which is modified by the ants when they visit the node. The pheromone concentration, $\phi i, j$ is an indication of the usage of this edge. An ant located in node *vi* uses pheromone $\phi i, j$ of node $vj \in Ni$ to compute the probability of node *vj* as next hop. *Ni* is the set of one-step neighbors of node *vi*.

$$pi,j = \begin{cases} \dfrac{\varphi i,j}{j \in Ni\ \varphi i,j} & \text{if } j \in Ni \\ 0 & \text{if } j \notin Ni \end{cases}$$

The transition probabilities *pi, j* of a node *vi* fulfill the constraint:

$$\sum_{j \in Ni} pi,j = 1, i\ \in [1,N]$$

During the route finding process, ants deposit pheromone on the edges. In the simple ant colony optimization meta-heuristic algorithm, the ants deposit a constant amount $\Delta\phi$ of pheromone. An ant changes the amount of pheromone of the edge *e*(*vi*, *vj*) when moving from node *vi* to node *vj* as follows:

$$\varphi i,j \coloneqq \varphi i,j + \Delta\varphi \qquad (1)$$

Like real pheromone the artificial pheromone concentration decreases with time to inhibit a fast convergence of pheromone on the edges. In the simple ant colony optimization meta-heuristic, this happens exponentially:

$$\varphi i,j \coloneqq (1-q).\varphi i,j, \qquad q \in (0,1] \qquad (2)$$

2.2 Particle Swarm Optimization (PSO) Model

The second example of a successful swarm intelligence model is Particle Swarm Optimization (PSO), which was introduced by Russell Eberhart, an electrical engineer, and James Kennedy, a social psychologist, in 1995. PSO was originally used to solve non-linear continuous optimization problems, but more recently it has been used in many practical, real-life application problems. For example, PSO has been successfully applied to track dynamic systems [9], evolve weights and structure of neural networks, analyze human tremor [11], register 3D-to-3D biomedical image [12], control reactive power and voltage [13], even learning to play games [14] and music composition [15]. PSO draws inspiration from the sociological behaviour associated with bird flocking. It is a natural observation that birds can fly in large groups with no collision for extended long distances, making use of their effort to maintain an optimum distance between themselves and their neighbours. This section presents some details about birds in nature and overviews their capabilities, as well as their sociological flocking behaviour.

2.2.1 Birds in Nature

Vision is considered as the most important sense for flock organization. The eyes of most birds are on both sides of their heads, allowing them to see objects on each side at the same time. The larger size of bird's eyes relative

to other animal groups is one reason why birds have one of the most highly developed senses of vision in the animal kingdom. As a result of such large sizes of bird's eyes, as well as the way their heads and eyes are arranged, most species of birds have a wide field of view. For example, *Pigeons* can see 300 degrees without turning their head, and *American Woodcocks* have, amazingly, the full 360-degree field of view. Birds are generally attracted by food; they have impressive abilities in flocking synchronously for food searching and long-distance migration. Birds also have an efficient social interaction that enables them to be capable of: (i) flying without collision even while often changing direction suddenly, (ii) scattering and quickly regrouping when reacting to external threats, and (iii) avoiding predators.

Birds Flocking Behaviour

The emergence of flocking and schooling in groups of interacting agents (such as birds, fish, penguins, etc.) have long intrigued a wide range of scientists from diverse disciplines including animal behaviour, physics, social psychology, social science, and computer science for many decades . Bird flocking can be defined as the social collective motion behaviour of a large number of interacting birds with a common group objective. The local interactions among birds (particles) usually emerge the shared motion direction of the swarm,. Such interactions are based on the *nearest neighbour principle* where birds follow certain flocking rules to 17 adjust their motion (i.e., position and velocity) based only on their nearest neighbours, without any central coordination. In 1986, birds flocking behavior were first simulated on a computer by Craig Reynolds. The pioneering work of Reynolds proposed three simple flocking rules to implement a simulated flocking behaviour of birds: (i) *flock centering* (flock members attempt to stay close to nearby flockmates by flying in a direction that keeps them closer to the centroid of the nearby flockmates), (ii) *collision avoidance* (flock members avoid collisions with nearby flockmates based on their relative position), and (iii) *velocity matching* (flock members attempt to match velocity with nearby flockmates) .

Although the underlying rules of flocking behavior can be considered simple, the flocking is visually complex with an overall motion that looks fluid yet it is made of discrete birds. One should note here that *collision avoidance* rule serves to establish the minimum required separation distance, whereas *velocity matching* rule helps to maintain such separation distance during flocking; thus, both rules act as a complement to each other. In fact, both rules together ensure that members of a simulated flock are free to fly without running into one another, no matter how many they are. It is worth mentioning that the three aforementioned flocking rules of Reynolds are generally known as *cohesion*, *separation*, and *alignment* rules in the literature. For example, according to the animal cognition and animal

behavior research, individuals of animals in nature are frequently observed to be attracted towards other individuals to avoid being isolated and to align themselves with neighbours. Reynolds rules are also compared to the *evaluation*, *comparison*, and *imitation* principles of the Adaptive Culture Model in the Social Cognitive Theory.

2.2.2 The Original PSO Algorithm

The original PSO was designed as a global version of the algorithm, that is, in the original PSO algorithm, each particle globally compares its fitness to the entire swarm population and adjusts its velocity towards the swarm's global best particle. There are, however, recent versions of local/topological PSO algorithms, in which the comparison process is locally performed within a predetermined neighbourhood topology. Unlike the original version of ACO, the original PSO is designed to optimize real-value continuous problems, but the PSO algorithm has also been extended to optimize binary or discrete problems. The original version of the PSO algorithm is essentially described by the following two simple *velocity* and *position* update equations, shown in 3 and 4 respectively.

$$vid(t+1) = vid(t) + c1\ \mathbf{R}1(pid(t) - xid(t)) + c2\ \mathbf{R}2\ (pgd(t) - xid(t)) \quad (3)$$

$$xid(t+1) = xid(t) + vid(t+1) \quad (4)$$

Where:

- *vid* represents the rate of the position change (velocity) of the *i*th particle in the *d*th dimension, and *t* denotes the iteration counter.
- *xid* represents the position of the *i*th particle in the *d*th dimension. It is worth noting here that **x***i* is referred to as the *i*th particle itself, or as a vector of its positions in all dimensions of the problem space. The n-dimensional problem space has a number of dimensions that equals to the numbers of variables of the desired fitness function to be optimized.
- *pid* represents the historically best position of the *i*th particle in the *d*th dimension (or, the position giving the best ever fitness value attained by **x***i*).
- *pgd* represents the position of the swarm's global best particle (**x*g***) in the *d*th dimension (or, the position giving the *global* best fitness value attained by *any* particle among the entire swarm).
- **R**1 and **R**2 are two n-dimensional vectors with random numbers uniformly selected in the range of [0.0, 1.0], which introduce useful randomness for the search strategy. It worth noting that each dimension has its own random number, *r*, because PSO operates on each dimension independently.

- *c1* and *c2* are positive constant weighting parameters, also called the *cognitive* and *social* parameters, respectively, which control the relative importance of particle‘s private experience versus swarm‘s social experience (or, in other words, it controls the movement of each particle towards its individual versus global best position). It is worth emphasizing that a single weighting parameter, c, called the acceleration constant or the learning factor, was initially used in the original version of PSO, and was typically set to equal 2 in some applications (i.e., it was initially considered that *c1 = c2 = c = 2*). But, to better control the search ability, recent versions of PSO are now using different weighting parameters which generally fall in the range of [0,4] with *c1 + c2 = 4* in some typical applications . The values of c1 and c2 can remarkably affect the search ability of PSO by biasing the new position of **x***i* toward its historically best position (its own private experiences, **Pi**), or the globally best position (the swarm‘s overall social experience, **Pg**):
    - High values of c1 and c2 can provide new positions in relatively distant regions of the search space, which often leads to a better global exploration, but it may cause the particles to diverge.
    - Small values of c1 and c2 limit the movement of the particles, which generally leads to a more refined local search around the best positions achieved.
    - When c1 > c2, the search behaviour will be biased towards particles historically best experiences.
    - When c1 < c2, the search behaviour will be biased towards the swarm‘s globally best experience.

The velocity update *equation* in (3) has three main terms: (i) The first term, *vid(t),* is sometimes referred to as *inertia*, *momentum* or *habit* . It ensures that the velocity of each particle is not changed abruptly, but rather the previous velocity of the particle is taken into consideration. That is why the particles generally tend to continue in the same direction they have been flying, unless there is a really major difference between the particle‘s current position from one side, and the particle‘s historically best position or the swarm‘s globally best position from the other side (which means the particle starts to move in the wrong direction).

This term has a particularly important role for the swarm‘s globally best particle, **x***g,* This is because if a particle, **x***i,* discovers a new position with a better fitness value than the fitness of swarm‘s globally best particle, then it becomes the global best (i.e., g←i). In this case, its historically best position, *pi,* will coincide with both the swarm‘s global best position, *pg,* and its own position vector, ***xi,*** in the next iteration (i.e., *pi* = ***xi*** = *pg*).

Therefore, the effect of last two terms in *equation* (3) will be no longer there, since in this special case $pid(t) - xid(t) = pig(t) - xid(t) = 0$.

This will prevent the global best particle to change its velocity (and thus its position), so it will keep staying at its same position for several iterations, as long as there was no way to offer an inertial movement and there has been no new best position discovered by another particle. Alternatively, when the previous velocity term is included in the velocity updating *equation* (3), the global best particle will continue its exploration of the search space using the inertial movement of its previous velocity. (ii) The second term, ***(pid(t) – xid(t))***, is the *cognitive*‖ part of the equation that implements a linear attraction towards the historically best position found so far by each particle. This term represents the private-thinking or the self-learning component from each particle‘s flying experience, and is often referred to as *local memory*, *self-knowledge*, *nostalgia* or *remembrance*. (iii) The third term, ***(pgd(t) – xid(t)),*** is the *social* part of the equation that implements a linear attraction towards the globally best position ever found by any particle . This term represents the experience-sharing or the group-learning component from the overall swarm‘s flying experience, and is often referred to as *cooperation*, *social knowledge*, *group knowledge* or *shared information*.

3. COMPARISON BETWEEN THE TWO SI MODELS

Despite both models are principally similar in their inspirational origin (the intelligence of swarms), and are based on nature-inspired properties, they are fundamentally different in the following aspects.

Table 1: Comparison of ACO and PSO

Criteria	ACO	PSO
Communication Mechanism	ACO uses an indirect communication mechanism among ants, called stigmergy, which means interaction through the environment	The communication among particles in PSO is rather direct without altering the environment.
Problem Types	ACO was originally used to solve combinatorial (discrete) optimization problems, but it was later modified to adapt continuous problems	PSO was originally used to solve continuous problems, but it was later modified to adapt binary/discrete optimization problems.
Problem Representation	ACO’s solution space is typically represented as a weighted graph, called construction graph.	PSO’s solution space is typically represented as a set of n-dimensional points
Algorithm Applicability	ACO is commonly more applicable to problems where	PSO is commonly more applicable to problems where

	source and destination are predefined and specific.	previous and next particle positions at each point are clear and uniquely defined.
Algorithm Objective	ACO's objective is generally searching for an optimal path in the construction graph.	PSO's objective is generally finding the location of an optimal point in a Cartesian coordinate system.
Examples of Algorithm Applications	Sequential ordering, scheduling, assembly line balancing, probabilistic TSP, DNA sequencing, 2D-HP protein folding, and protein–ligand docking.	Track dynamic systems, evolve NN weights, analyze human tremor, register 3D-to-3D biomedical image, control reactive power and voltage, and even play games .

4. CONCLUSION

The ACO and PSO can be analyzed for future enhancement such that new research could be focused to produce better solution by improving the effectiveness and reducing the limitations. More possibilities for dynamically determining the best destination through ACO can be evolved and a plan to endow PSO with fitness sharing aiming to investigate whether this helps in improving performance. In future the velocity of each individual must be updated by taking the best element found in all iterations rather than that of the current iteration only.

REFERENCES

[1] R Kumar, M K Tiwari and R Shankar, " Scheduling of flexible manufacturing systems: an ant colony optimization approach", Proceedings Instn Mech Engrs Vol. 217, Part B: J Engineering Manufacture, 2003,pp 1443-1453.

[2] Kuan Yew Wong, Phen Chiak See, " A New minimum pheromone threshold strategy(MPTS) for Max-min ant system ", Applied Soft computing, Vol. 9, 2009, pp 882-888.

[3] David C Mathew, "Improved Lower Limits for Pheromone Trails in ACO", G Rudolf et al(Eds), LNCS 5199, pp 508-517, Springer Verlag, 2008.

[4] Laalaoui Y, Drias H, Bouridah A and Ahmed R B, " Ant Colony system with stagnation avoidance for the scheduling of real time tasks", Computational Intelligence in scheduling, IEEE symposium, 2009, pp 1-6.

[5] E Priya Darshini, " Implementation of ACO algorithm for EDGE detection and Sorting Salesman problem",International Journal of Engineering science and Technology, Vol 2, pp 2304-2315, 2010

[6] Alaa Alijanaby, KU Ruhana Kumahamud, Norita Md Norwawi, "Interacted Multiple Ant a. Colonies optimization Frame work: an experimental study of the evaluation and the

exploration techniques to control the search stagnation", International Journal of Advancements in computing Technology Vol. 2, No 1, March 2010, pp 78-85

[7] Raka Jovanovic and Milan Tuba, " An ant colony optimization algorithm with improved pheromone correction strategy for the minimum weight vertex cover problem", Elsvier,Applied Soft Computing, PP 5360-5366,2011.

[8] Zar Ch Su Su Hlaing, May Aye Lhine, " An Ant Colony Optimisation Algorithm for solving Traveling Salesman Problem", International Conference on Information Communication and Management(IPCSIT), Vol,6, pp 54-59, 2011.

[9] D. Karaboga, "An Idea Based On Honey Bee Swarm for Numerical Optimization", Technical Report-TR06,Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[10] M. Bakhouya and J. Gaber, "An Immune Inspired-based Optimization Algorithm: Application to the Traveling Salesman Problem, Advanced Modeling and Optimization", Vol. 9, No. 1, pp. 105-116, 2007.

[11] K. N. Krishnanand and D. Ghose, "Glowworm Swarm Optimization for searching higher dimensional spaces". In: C. P. Lim, L. C. Jain, and S. Dehuri (eds.) Innovations in Swarm Intelligence. Springer, Heidelberg, 2009.

[12] M. P. Wachowiak, R. Smolíková, Y. Zheng, J. M. Zurada, and A. S. Elmaghraby, "An approach to multimodal biomedical image registration utilizing particle swarm optimization", IEEE Transactions on Evolutionary Computation, 2004.

[13] L. Messerschmidt, A. P. Engelbrecht, "Learning to play games using a PSO-based competitive learning approach", IEEE Transactions on Evolutionary Computation, 2004.

[14] T. Blackwell and P. J. Bentley, "Improvised music with swarms, In David B. Fogel, Mohamed A. El-Sharkawi, Xin Yao, Garry Greenwood, Hitoshi Iba, Paul Marrow, and Mark Shackleton (eds.), Proceedings of the 2002 Congress on Evolutionary Computation CEC 2002, pages 1462–1467, IEEE Press, 2002.