Comparison of PSO and ABC: From A Viewpoint of Learning

Guo-sheng HAO¹, Gai-ge WANG¹, Zhao-jun ZHANG² and De-xuan ZOU²

¹School of Computer Science & Technology, Jiangsu Normal University, Xuzhou, 221116, China
²School of electrical engineering & automation, Jiangsu Normal University, Xuzhou, 221116, China

*Corresponding author

Keywords: Learning mechanism, PSO, ABC, Learning rules, Exploration, Exploitation.

Abstract. The learning mechanisms in Partial Swarm Optimization (PSO) and Artificial Bee Colony (ABC) algorithm are studied. Four basic learning elements are considered, including learning subject, learning object, learning result and learning rules. Both PSO and ABC generate new solutions by learning to explore/exploit promising subspace. For the solution generation operators in each algorithm, we study the learning mechanism and analyze their exploration and exploitation ability. This study gives more insights on the similarity and differences between PSO and ABC.

Introduction

For many decades, researchers have been developing ever more sophisticated nature-inspired metaheuristic algorithms (NIOAs)[1] to solve hard optimization problems. Among NIOAs, Genetic Algorithm (GA) [2] was firstly introduced. Since then, various other optimization algorithms have been proposed such as Particle Swarm Optimization (PSO)[3] [4], Artificial Bee Colony (ABC) algorithm [5] and so on.

Faced with many NIOAs, there are some questions that are often asked by researchers and application researchers such as: (1) Learning is at the core of intelligence. How does learning work in NIOAs? (2) How does learning guide the exploitation and exploration? (3) What are the similarities and differences among NIOAs, such as between PSO and ABC? Different NIOAs are inspired by different natural phenomena and different NIOAs may have different learning strategies or mechanisms to update their population.

There are, of course, many learning methods or many ways to generate new solutions. Regardless of the specific ways, it is generally agreed that NIOAs utilize operators to carry out learning or generate new solutions.

Almost all NIOAs are implemented with the flowchart shown in Figure 1 (a). Therefore, different NIOAs can be compared briefly from the viewpoint of new solution generation, e.g., PSO and ABC in Figure 1 (b) and (c) respectively.

![Figure 1. Flowchart of optimization algorithms, PSO and ABC.](image-url)
The following study is based on the maximum optimization case. It should be noted that it is not the goal of this paper to declare one algorithm or the other are somehow better. Instead, the goal is to provide the comparison among NIOAs from the viewpoint of learning mechanism behinds the operators in NIOAs.

The aim of learning is to explore or exploit the promising search space. Therefore, we can measure learning results by measuring the exploration and exploitation. How to measure exploration and exploitation is an open question [6]. Here we measure them by borrowing the measurement of the modal or peaks of the optimized problem. If the learning results \{X_{new}\} have the same peak with learning subjects \{X_S\} (or learning objects \{X_O\}, if applicable), we call that the learning is exploitation. If the learning results \{X_{new}\} have the different peak with learning subjects \{X_S\} (or learning objects \{X_O\}, if applicable), we call that the learning is exploration.

Swarm intelligence algorithm has a key feature as self-organization which results collective behavior by means of local interactions among simple swarm agent [7]. The swarm agent can be birds, fishes, bees, krill, cuckoos, fireflies, wolves and so on.

**Analysis of Learning Mechanism of Solution Generation in Partial Swarm Optimization**

Particle swarm optimization is a swarm intelligence technique developed by Eberhart and Kennedy [3, 4], based on the social behaviour of bird flocking and fish schooling. For every swarm, it flies (or swims or moves) with a variable velocity to a promising location where they can profit from the discoveries or previous experiences of all other members of the school during the search for food.

Different from Genetic Algorithm (GA), PSO does not adopt an explicit selection function. The absence of a selection mechanism in PSO is compensated by the use of leaders to guide the search. In addition, different from GA, there is no notion of offspring generation in PSO [8]. The location of a particle is treated as a solution in search space. The update of solution is the update of the position of a particle. Rather than use crossover or mutation, it uses real-number randomness and global communication among the swarm particles [9].

PSO has been shown as a very effective optimizer, especially in large convoluted search space [10]. Generally, a particle update includes two properties update: velocity update and location update.

We will analyse the learning mechanism in the update of velocity and location in PSO. The partial flowchart of PSO is shown in Figure 1(b), and for the other part, one is referred to Figure 1.

In PSO, the best previous position of a solution is recorded and represented as \(X_i^{*} = (x_i^{*(1)}, x_i^{*(2)}, \ldots, x_i^{*(d)})\). And the index of the best particle among all the particles in the population is represented by the symbol \(g\) and the corresponding position of this solution is represented as \(X_g^{*} = (x_g^{*(1)}, x_g^{*(2)}, \ldots, x_g^{*(d)})\).

**Velocity Update.** Denote the velocity of a particle as \(\Delta \mathbf{X}_i = (\Delta x_i^{(1)}, \Delta x_i^{(2)}, \ldots, \Delta x_i^{(d)})\). Then the learning subject and objects in velocity update is: \(X_S = X_i, \{X_O\} = \{X_i, X_i^*, X_g^*\}\), and the learning rule is: \(\Delta x_i^{(k)} = \omega_1 \Delta x_i^{(k)} + \omega_2 \text{rand}_1(x_i^{*(k)} - x_i^{(k)}) + \omega_3 \text{rand}_2(x_g^{*(k)} - x_i^{(k)})\), where \(\omega_1, \omega_2, \text{ and } \omega_3\) are weights value and \(\text{rand}_1, \text{rand}_2\) are two random values. Generally, there is \(\omega_1 > 0, \omega_2 > 0, \omega_3 > 0 \text{ and } \omega_2 + \omega_3 \leq 2\) [11]. In next iteration, \(\Delta x_i^{(k)}\) uses \(\Delta x_{new,1}^{(k)}\) as its new value.

The parameter \(\omega_1\) is also called as the inertial weight to control the impact of the previous history of velocities on the current velocity. In the velocity update, a solution can learn not only from its own previous best solution but also from the global best solution. By learning, \(X_i\) adjusts its motion toward \(X_i^*\) and \(X_g^*\).

Figure 2 (a) shows velocity update and the shaded part is the value area of \(\omega_2 \text{rand}_1(x_i^{*(k)} - x_i^{(k)}) + \omega_3 \text{rand}_2(x_g^{*(k)} - x_i^{(k)})\).

**Location Update.** The learning subject and objects of location update in PSO are: \(X_S = X_i \in P, \{X_O\} = \{X_i^*, X_g^*\}\). The learning rule is: \(x_i^{(k)} = x_i^{(k)} + \Delta x_i^{(k)}\), where \(\Delta x_{new,1}^{(k)}\) is the new velocity of a solution.
An example of a 2-dimensional case and the process for location update is shown in Figure 2 (b)-(d). Figure 2 (b) shows the relationship of learning results, learning subject and learning objects. Figure 2 (c) shows the exploration effect of solution update and Figure 2 (d) shows the exploitation effect of solution update.

Substitute $\Delta x^{(k)}_{new}$ into $x^{(k)}_{new}$, we can get an integrated learning rule $L(\cdot)$ as:

$$x^{(k)}_{new,2} = x^{(k)}_{i} + \omega_1 \Delta x^{(k)}_{i} + \omega_2 rand_1(x^{(k)}_{i} - x^{(k)}_{i}) + \omega_3 rand_2(x^{(k)}_{i} - x^{(k)}_{i}).$$

### Analysis of Learning Mechanism of Solution Generation in Artificial Bee Colony Algorithm

Several approaches have been proposed to model the collective intelligence of honeybee [5] [12-19]. The honeybee swarms model consists of three components: food sources, employed foragers and unemployed foragers [20]. The food sources are corresponding to the solutions in the search space. Employed foragers are associated with a particular solution, which they are currently exploiting or are "employed" at. Unemployed foragers are continually at look out for a food source to exploit. There are two types of unemployed foragers: scouts and onlookers. Scouts searching the environment surrounding the nest for new food sources and onlookers waiting in the nest and establishing a food source through the information shared by employed foragers [20]. The mean number of scouts averaged over conditions is about 5-10% of other bees [21].

In Artificial Bee Colony (ABC) algorithm, there are two modes of the behavior: the recruitment to a rich nectar source and the abandonment of a poor nectar source. Initially, scout bees discover all food source positions. Thereafter, the nectar of food sources are exploited by employed bees and onlooker bees. Then, the employed bee which was exploiting the exhausted food source becomes a scout bee in search of further food sources once again [20].

The partial flowchart is shown in Figure 1 (c), and for the other part, one is referred to Figure 1 (a).

We study the learning mechanism in ABC in the order of employed bee update, onlooker selection and scout randomly generated solutions.

**Employed Bee Update.** The employed bees carry out exploitation by learning from other bees including onlookers, scout and other employed bees. Therefore, the learning subject and object are $X_S = X_i, X_j \in P, X_O = X_j, i \neq k, X_i \in P$, where $j$ is randomly selected in the set $\{1, 2, \ldots, |P|\}$. And the learning rule $L(\cdot)$ is: $x^{(k)}_{new,1} = x^{(k)}_{i} + \text{rand}(x^{(k)}_{i} - x^{(k)}_{j})$, where $\text{rand}$ is a random value in [-1, 1].

An example of a 2-dimensional case and the process for employed bee update is shown in Figure 2, where the shaded area is location for all the learning results for $\text{rand}$ is in [-1, 1]. The relationship of learning subject, learning object and learning results are shown in Figure 2 (a). And the exploration and exploitation effect are shown in Figure 2 (b) and (c).

After the components update, the solution will be determined by competing as $X_{new, 2} = \{ X_{new, 1} \times X_i | X_i \neq X_{new, 1} \} = |P|$. 
Onlookers update. Onlookers exchange their information with employed bees. An onlooker chooses the solution depending on the probability value associated with fitness. The choosing of solutions support the mechanism of learning from promising solution [22] and it is similar to the selection operator in GA. The learning rule $L(\cdot)$ is $\{X_{\text{new,2}}\} = \bigcup_{i=1}^{[P]} \{\omega_i X_i\}$, $X_i \in \{X_{\text{new,2}}\}$, $\omega_i \in \{0, 1, \ldots, |P|\}$.

The probability for a solution $X_i$ to be selected by onlookers is proportion to its fitness: $\frac{f(X_i)}{\sum_{x_j \in \{X_{\text{new,2}}\}} f(x_j)}$. The exploitative effect is also the same as selection operator in GA, and we omit it here.

Scout Update. The scout produces random solution. If the solution that employed bee manipulated cannot be improved further through a predetermined number of iterations, the solution will be replaced by the scout produced solution. The scout helps ABC to avoid the risk of becoming trapped by a local optimum.

Conclusion
From the viewpoint of learning mechanism, we study the similarities and differences between PSO and ABC. This will help theory and application researcher to understand these algorithms better.

Acknowledgement
This research was financially supported by the National Science Foundation of China under Grants No. 61673196, 61503165, 61403174.

References


