Chapter 1
Unified Swarm Intelligence Algorithms

Yuhui Shi
Southern University of Science and Technology (SUSTech), China

ABSTRACT

In this chapter, the necessity of having developmental learning embedded in a swarm intelligence algorithm is confirmed by briefly considering brain evolution, brain development, brainstorming process, etc. Several swarm intelligence algorithms are looked at from a developmental learning perspective. A framework of a developmental swarm intelligence algorithm, which contains capacity developing stage and capability learning stage, is further given to help understand developmental swarm intelligence (DSI) algorithms, and to guide to design and/or implement any new developmental swarm intelligence algorithm and/or any developmental evolutionary algorithm. Following DSI, innovation is discussed and an innovation-inspired optimization (IO) algorithm is designed and developed. Finally, by combing the DSI and IO algorithm together, a unified swarm intelligence algorithm is proposed, which contains capacity developing stage and capability learning stage and with three search operators in its capability learning stage to mimic the three levels of innovations.

INTRODUCTION

Can a swarm intelligence algorithm develop its learning capacity that can better solve an optimization problem which is unknown at the algorithm's design or implementation time? In the swarm intelligence research field, we are required to solve different types of optimization problems under different environments. For example, there are single objective optimization problems, multi-objective optimization problems, constrained optimization problems, combinatorial optimization problems, *etc*.; there are optimization problems under fixed environment, dynamically changing environment, unknown environment, *etc*. As claimed in no-free-lunch theory (Wolpert & Macready, 1997), there is no single algorithm that will work the best for all different problems. That is to say, one algorithm can be better for one kind of problems, but may be worse for other kinds of problems. It usually is not an easy, if not impossible, job to find the best algorithm for solving one kind of problems, especially when we have no prior knowledge

DOI: 10.4018/978-1-5225-5134-8.ch001

about the problem and the environment the problem is in. An ideal optimization algorithm should have the ability to change itself to have the suitable capacity to learn and solve the problem to be solved under its own environment, that is to say, it should be able to develop its own learning capacity or learning potential which has special connection with the problem and its environment, therefore, to enable the algorithm to better learn and solve the problem.

Researches on optimization algorithms have been around for many years because many real-world problems can eventually be represented or modelled as optimization problems which then require optimization algorithms to solve or to find solutions. Traditionally, hill-climbing algorithms are commonly used to solve optimization problems which usually require optimization problems to be able to be represented by mathematic functions which are further required to be continuous and differentiable. One commonly used hill-climbing algorithm is the steepest descent approach (Battiti, 1992). For an optimization problem, if it can be represented by a continuous and differentiable convex function, then the steepest descent approach can always find its global optimal solution; if the problem can't be represented by a convex function, then the steepest descent approach will find a solution which may or may not be a global optimal solution (point), and in general, it will only find a local optimal solution which may or may not be a good enough solution. What kind of local optimal solution it may find depends on the initial starting solution. Therefore, hill-climbing algorithms are usually called local search algorithms (Hoos & Stutzle, 2005). A hill-climbing algorithm has the capability to find an optimal solution, but whether it will find or not depends on its initial starting solution. To overcome this issue of failing into local optimum, some techniques have been added to modify hill-climbing algorithms to make them to have more potential to avoid local optimum and eventually find at least a good enough solution. These modified algorithms include stochastic gradient descent algorithms (Gardner, 1984), random walk algorithms (Grady, 2006), and simulated annealing algorithms (Granville *et al.*, 1994).

In order to overcome or remove the limitation that an optimization problem needs to be represented or modelled by a continuous and differentiable function and to improve its possibility of finding better solutions, population-based heuristic algorithms were proposed and studied, which can be applied to solve optimization problems which are not required to be represented by continuous and differentiable function, instead the requirement is lessened to be that any solution to the problem can be evaluated. Commonly used population-based heuristic algorithms include genetic algorithms (Holland, 1975), evolutionary programming (Fogel, 1962), genetic programming (Koza, 1992), evolution strategy (Rechenberg, 1973), and swarm intelligence algorithms (Eberhart & Shi, 2007). So far, most researches on these population-based heuristic algorithms focus on their search capability or learning capability.

To further improve performance of population-based heuristic algorithms, researches have been conducted to combine different algorithms to take advantages from each algorithm. For example, one algorithm may be good at exploration while the other may be good at exploitation (Zhu *et al.*, 2011), therefore, it is expected to have better balanced exploration and exploitation through combination. Another research trend is to adapt an algorithm's parameters to make the algorithm to dynamically fit for its search environment which usually is nonlinearly and dynamically changed (Zhan *et al.*, 2009). In general, the search process of an algorithm will be in different search "state" which requires the algorithm to be in different "potential" or to have different search capacity in addition to its learning capability. A good optimization (search) algorithm should have both learning capability and also its ability to fit into different search "state" or learning capacity (or potential). It is important for an optimization algorithm to have learning capability while it is critical for it to have learning capacity or learning potential. An optimization algorithm with both learning capability and learning capacity can be called a developmental