Firefly Algorithms for Multimodal Optimization

Xin-She Yang

Department of Engineering, University of Cambridge,
Trumpington Street, Cambridge CB2 1PZ, UK
xy227@cam.ac.uk

Abstract. Nature-inspired algorithms are among the most powerful algorithms for optimization. This paper intends to provide a detailed description of a new Firefly Algorithm (FA) for multimodal optimization applications. We will compare the proposed firefly algorithm with other metaheuristic algorithms such as particle swarm optimization (PSO). Simulations and results indicate that the proposed firefly algorithm is superior to existing metaheuristic algorithms. Finally we will discuss its applications and implications for further research.

1 Introduction

Biologically inspired algorithms are becoming powerful in modern numerical optimization [1, 2, 4, 6, 9, 10], especially for the NP-hard problems such as the travelling salesman problem. Among these biology-derived algorithms, the multi-agent metaheuristic algorithms such as particle swarm optimization form hot research topics in the start-of-the-art algorithm development in optimization and other applications [1, 2, 9].

Particle swarm optimization (PSO) was developed by Kennedy and Eberhart in 1995 [5], based on the swarm behaviour such as fish and bird schooling in nature, the so-called swarm intelligence. Though particle swarm optimization has many similarities with genetic algorithms, but it is much simpler because it does not use mutation/crossover operators. Instead, it uses the real-number randomness and the global communication among the swarming particles. In this sense, it is also easier to implement as it uses mainly real numbers.

This paper aims to introduce the new Firefly Algorithm and to provide the comparison study of the FA with PSO and other relevant algorithms. We will first outline the particle swarm optimization, then formulate the firefly algorithms and finally give the comparison about the performance of these algorithms. The FA optimization seems more promising than particle swarm optimization in the sense that FA can deal with multimodal functions more naturally and efficiently. In addition, particle swarm optimization is just a special class of the firefly algorithms as we will demonstrate this in this paper.

O. Watanabe and T. Zeugmann (Eds.): SAGA 2009, LNCS 5792, pp. 169–178, 2009.

2 Particle Swarm Optimization

2.1 Standard PSO

The PSO algorithm searches the space of the objective functions by adjusting the trajectories of individual agents, called particles, as the piecewise paths formed by positional vectors in a quasi-stochastic manner [5, 6]. There are now as many as about 20 different variants of PSO. Here we only describe the simplest and yet popular standard PSO.

The particle movement has two major components: a stochastic component and a deterministic component. A particle is attracted toward the position of the current global best $\mathbf{g}^*$ and its own best location $\mathbf{x}_i^*$ in history, while at the same time it has a tendency to move randomly. When a particle finds a location that is better than any previously found locations, then it updates it as the new current best for particle i. There is a current global best for all n particles. The aim is to find the global best among all the current best solutions until the objective no longer improves or after a certain number of iterations.

For the particle movement, we use $\mathbf{x}_i^*$ to denote the current best for particle i, and $\mathbf{g}^* \approx \min$ or $\max\{f(\mathbf{x}_i)\}(i = 1, 2, ..., n)$ to denote the current global best. Let $\mathbf{x}_i$ and $\mathbf{v}_i$ be the position vector and velocity for particle i, respectively. The new velocity vector is determined by the following formula

$$\mathbf{v}_i^{t+1} = \mathbf{v}_i^t + \alpha\epsilon_1 \odot (\mathbf{g}^* - \mathbf{x}_i^t) + \beta\epsilon_2 \odot (\mathbf{x}_i^* - \mathbf{x}_i^t). \tag{1}$$

where ϵ_1 and ϵ_2 are two random vectors, and each entry taking the values between 0 and 1. The Hadamard product of two matrices $\mathbf{u} \odot \mathbf{v}$ is defined as the entrywise product, that is $[\mathbf{u} \odot \mathbf{v}]_{ij} = u_{ij}v_{ij}$. The parameters α and β are the learning parameters or acceleration constants, which can typically be taken as, say, $\alpha \approx \beta \approx 2$. The initial values of $\mathbf{x}_i^{t=0}$ can be taken as the bounds or limits $a = \min(x_j)$, $b = \max(x_j)$ and $\mathbf{v}_i^{t=0}= 0$. The new position can then be updated by

$$\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \mathbf{v}_i^{t+1}. \tag{2}$$

Although $\mathbf{v}_i$ can be any values, it is usually bounded in some range $[0, \mathbf{v}_{max}]$.

There are many variants which extend the standard PSO algorithm, and the most noticeable improvement is probably to use inertia function $\theta(t)$ so that $\mathbf{v}_i^t$ is replaced by $\theta(t)\mathbf{v}_i^t$ where θ takes the values between 0 and 1. In the simplest case, the inertia function can be taken as a constant, typically $\theta \approx 0.5 \sim 0.9$. This is equivalent to introducing a virtual mass to stabilize the motion of the particles, and thus the algorithm is expected to converge more quickly.

3 Firefly Algorithm

3.1 Behaviour of Fireflies

The flashing light of fireflies is an amazing sight in the summer sky in the tropical and temperate regions. There are about two thousand firefly species, and