Neurocomputing ■ (■■■■) ■■■–■■■

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A hybrid fireworks optimization method with differential evolution operators

Yu-Jun Zheng [a,*], Xin-Li Xu [a], Hai-Feng Ling [b], Sheng-Yong Chen [a]

[a] College of Computer Science & Technology, Zhejiang University of Technology, Hangzhou 310023, China
[b] Department of Mechanical Engineering, PLA University of Science & Technology, Nanjing 210007, China

ARTICLE INFO

Article history:
Received 7 April 2012
Received in revised form
26 August 2012
Accepted 27 August 2012

Keywords:
Fireworks algorithm (FA)
Differential evolution (DE)
Global optimization
Hybrid

ABSTRACT

Fireworks algorithm (FA) is a relatively new swarm-based metaheuristic for global optimization. The algorithm is inspired by the phenomenon of fireworks display and has a promising performance on a number of benchmark functions. However, in the sense of swarm intelligence, the individuals including fireworks and sparks are not well-informed by the whole swarm. In this paper we develop an improved version of the FA by combining with differential evolution (DE) operators: mutation, crossover, and selection. At each iteration of the algorithm, most of the newly generated solutions are updated under the guidance of two different vectors that are randomly selected from highly ranked solutions, which increases the information sharing among the individual solutions to a great extent. Experimental results show that the DE operators can improve diversity and avoid prematurity effectively, and the hybrid method outperforms both the FA and the DE on the selected benchmark functions.

1. Introduction

The complexity of real-world engineering optimization problems gives rise to various kinds of metaheuristics that use stochastic techniques to effectively explore the search space for a global optimum. In particular, metaheuristics based on swarm intelligence (e.g., [1–7]), which simulates a population of simple individuals evolving their solutions by interacting with one another and with the environment, have shown promising performance on many difficult problems and have become a very active research area in recent years.

Fireworks algorithm (FA) is a relatively new global optimization method originally proposed by Tan and Zhu [7]. Inspired by the phenomenon of fireworks explosion, the algorithm selects in the search space a certain number of locations, each for exploding a firework to generate a set of sparks. The fireworks and sparks of high quality are chosen as the locations for the next generation's fireworks, and the evolutionary process continues until a desired optimum is obtained, or the stopping criterion is met. Numerical experiments on a number of benchmark functions show that the FA can converge to a global optimum with a much smaller number of function evaluations than that of typical particle swarm optimization (PSO) algorithms including [1,8].

In the standard FA, the convergence speed is accelerated by "good" fireworks that generate more sparks within smaller explosion areas, and the search diversity is improved by "bad" fireworks that generate fewer sparks within larger explosion areas. However, to some extent, such a diversification mechanism is not very flexible and, in particular, it does not utilize more information about other quality solutions in the swarm. That is, in the sense of swarm intelligence, the individuals (fireworks and sparks) are not well-informed by the whole swarm.

Inspired by this observation, we develop an improved fireworks optimization method by combining with differential evolution (DE) operators: mutation, crossover, and selection [9]. At each iteration of the algorithm, these operators are applied to guide the generation of new solutions, which improves the diversity of the swarm and avoids being trapped in local optima too early. Experiments on selected benchmark functions show that the well-informed fireworks and sparks can improve the performance of the FA to a great extent.

The remainder of this paper is structured as follows: Section 2 briefly describes the FA algorithm and the DE algorithm, Section 3 proposes the framework of our hybrid FA method, Section 4 presents the computational experiments, Section 5 analyzes and discusses the experimental results, and Section 6 makes the conclusion.

2. Backgrounds

2.1. Fireworks algorithm

The FA proposed in [7] is a global optimization algorithm simulating the explosion process of fireworks, where an explosion

* Corresponding author. Tel.: +86 571 85290085.
E-mail addresses: yujun.zheng@computer.org (Y.-J. Zheng), xxl@zjut.edu.cn (X.-L. Xu), hf.ling@ymail.com (H.-F. Ling), sy@ieee.org (S.-Y. Chen).

http://dx.doi.org/10.1016/j.neucom.2012.08.075
0925-2312/

can be viewed as a search in the local space around the location of a firework. In the original FA, the number of sparks and the amplitude of explosion for each firework x_i are respectively defined as follows:

$$s_i = m \cdot \frac{f_{max} - f(x_i) + \epsilon}{\sum_{j=1}^{p} (f_{max} - f(x_j)) + \epsilon} \tag{1}$$

$$A_i = \hat{A} \cdot \frac{f(x_i) - f_{min} + \epsilon}{\sum_{j=1}^{p} (f(x_j) - f_{min}) + \epsilon} \tag{2}$$

where m is a parameter for controlling the total number of sparks generated by the fireworks, $\hat{A}$ is the maximum explosion amplitude, p is the size of the swarm, f_{max} and f_{min} are respectively the maximum and minimum objective values among the p fireworks, and ϵ is a small constant to avoid zero-division-error.

To avoid overwhelming effects of splendid fireworks, lower and upper bounds are defined for s_i such that

$$s_i = \begin{cases} s_{min} & \text{if } s_i < s_{min} \\ s_{max} & \text{else if } s_i > s_{max} \\ s_i & \text{else} \end{cases} \tag{3}$$

For a D-dimensional problem, the location of each spark x_j generated by x_i can be obtained by randomly setting z directions ($z < D$), and for each dimension k setting the component x_j^k based on x_i^k ($1 \le j \le s_i, 1 \le k \le z$). There are two ways for setting x_j^k. For most sparks, a displacement $h_k = A_i \cdot rand(-1, 1)$ is added to x_i^k,i.e.,

$$x_j^k = x_i^k + A_i \cdot rand(-1, 1) \tag{4}$$

To keep the diversity, for a few specific sparks, an explosion coefficient based on Gaussian distribution is applied to x_i^k such that

$$x_j^k = x_i^k \cdot Gaussian(1, 1) \tag{5}$$

In both the ways, if the new location falls out of the search space, it is mapped to the search space as follows:

$$x_j^k = x_{min}^k + |x_j^k| \% (x_{max}^k - x_{min}^k) \tag{6}$$

where % denotes the modulo operator for floating-point numbers, as defined in most computer languages.

At each iteration of the FA, among all the current sparks and fireworks, the best location is always selected as a firework of the next generation. After that, $p-1$ fireworks are selected with probabilities proportional to their distance to other locations. The general framework of the FA is described in Algorithm 1.

In the FA, sparks suffer the power of explosion and thus move along z directions simultaneously, which makes the algorithm converge very fast. Two types of spark generation methods and the specific selection process for locations also endue the FA with the capability of avoiding premature convergence. The advantages of the FA over the standard PSO and improved PSO algorithms have also been demonstrated by experiments on a number of benchmark functions [7].

Algorithm 1. The standard fireworks algorithm.

set the algorithm parameters p, s_{min}, s_{max}, $\hat{A}$, and $\hat{m}$;
randomly initialize a swarm S of p fireworks;
while (stop criteria is not met) **do**
  let R be the empty set of sparks;
  foreach firework $x_i \in S$ **do**
    calculate s_i for x_i according to Eqs. (1) and (3);
    calculate A_i for x_i according to Eq. (2);
    for $j = 1$ to s_i **do**
      yield a spark x_j;
      let $z = round(D \cdot rand(0, 1))$;
      for $k = 1$ to z **do** set x_j^k according to Eqs. (4) and (6);
      $R = R \cup \{x_j\}$;
  randomly select a set P of $\hat{m}$ fireworks from S;
  foreach firework $x_i \in P$ **do**
    yield a spark x_j;
    let $z = round(D \cdot rand(0, 1))$;
    for $k = 1$ to z **do** set x_j^k according to Eqs. (5) and (6);
    $R = R \cup \{x_j\}$;
  $R = R \cup S$;
  let $gbest$ be the best location among R, and set $S = \{gbest\}$;
  Add to S other $p-1$ locations selected from R based on distance probabilities;
end

2.2. Difference evolution

Introduced by Storn and Price [9], DE is an efficient evolutionary algorithm that simultaneously evolves a population of solution vectors. But unlike the genetic algorithm (GA) [10], DE uses floating-point vectors and does not employ some probability density functions for vector reproduction. Specifically, DE generates a mutant vector $\mathbf{v}_i$ for each vector $\mathbf{x}_i$ in the population by adding the weighted difference between two randomly selected vectors to a third one:

$$\mathbf{v}_i = \mathbf{x}_{r_1} + \gamma(\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) \tag{7}$$

where random indexes $r_1, r_2, r_3 \in \{1, 2, \ldots, p\}$ and coefficient $\gamma > 0$.

A trial vector $\mathbf{u}_i$ is then generated by using the crossover operator which mixes the components of the mutant vector and the original one, where each jth component of $\mathbf{u}_i$ is determined as follows:

$$\mathbf{u}_i^j = \begin{cases} \mathbf{v}_i^j & \text{if } rand(0,1) < c_r \text{ or } j = r(i) \\ \mathbf{x}_i^j & \text{else} \end{cases} \tag{8}$$

where c_r is the crossover probability ranged in $(0, 1)$ and $r(i)$ is a random integer within $(0, p]$ for each i.

In the last step of each iteration, the selection operator chooses the better one for the next generation by comparing $\mathbf{u}_i$ with $\mathbf{x}_i$:

$$\mathbf{x}_i = \begin{cases} \mathbf{u}_i & \text{if } f(\mathbf{u}_i) \le f(\mathbf{x}_i) \\ \mathbf{x}_i & \text{else} \end{cases} \tag{9}$$

By computing the difference between two individuals randomly chosen from the population, the DE is actually estimating the gradient in that zone (rather than in a point). The mutation operator makes the DE capable of self-adapting both the step sizes and the step direction, and local criterion of the selection operator is also efficient and fast [11]. Generally, these features make the DE converge faster and with more certainty than many other heuristic methods.

3. The hybrid fireworks optimization method

For a D-dimensional optimization, the fitness value of a solution is determined by values of all components, and a solution that has discovered the region corresponding to the global optimum in some dimensions may have a low fitness value because of the poor quality in the other dimensions [12]. Thus, many population-based optimization methods, including DE, comprehensive learning PSO [12] and fully informed PSO [13], enable the individuals to make use of the beneficial information in the swarm more effectively to generate better quality solutions.

In the standard FA, after obtaining the set R of all fireworks and sparks, the locations for new fireworks are selected based on distance to other locations in R so as to keep diversity of the