ELSEVIER

Journal of Systems Engineering and Electronics

Vol. 18, No. 3, 2007, pp.650–654

Available online at www.sciencedirect.com

ScienceDirect

www.sciencedirect.com/science/journal/10044132

The self-organizing worm algorithm

Zheng Gaofei[1,2], *Wang Xiufeng*[2] & *Zhang Yanli*[3]

1. Dept. of Mechanics, Tianjin Polytechnic Univ., Tianjin 300160, P. R. China;

2. Information Technology Science Coll., Nankai Univ., Tianjin 300071, P. R. China;

3. Information & Communitiaon of Engineering School, Tianjin Polytechnic Univ., Tianjin 300160, P. R. China

(Received May 19, 2006)

Abstract: A new multi-modal optimization algorithm called the self-organizing worm algorithm (SOWA) is presented for optimization of multi-modal functions. The main idea of this algorithm can be described as follows: disperse some worms equably in the domain; the worms exchange the information each other and creep toward the nearest high point; at last they will stop on the nearest high point. All peaks of multi-modal function can be found rapidly through studying and chasing among the worms. In contrast with the classical multi-modal optimization algorithms, SOWA is provided with a simple calculation, strong convergence, high precision, and does not need any prior knowledge. Several simulation experiments for SOWA are performed, and the complexity of SOWA is analyzed amply. The results show that SOWA is very effective in optimization of multi-modal functions.

Keywords: control theory, multi-modal optimization algorithm, self-organizing worm algorithm; unit

1. Introduction

Because of the limitation of impersonality, the global optimal solution can not be reached in many optimization problems. Therefore, not only the global optimal solution but also some local optimal solutions have to be searched in the domain. These local solutions can supply a variety of selections for the decision maker. This problem is named multi-modal function optimization. Therefore, searching both the global and the local solutions has been a hotspot in multi-modal optimization.

Multi-modal genetic algorithms (MGA) is a special genetic algorithm (GA) for searching the multi-peak of the multi-modal functions. In 1975, De Jong[1] introduced crowding factor into MGA. In 1985, Perry[2] applied the habitat theory on GA. In 1975, Holland[3] introduced the Niche theory initially; Goldberg and Richardson[4] realized the Niche theory in 1987 by a combined method of sharing and restricted mating. In 1994, William M.Spears[5] put forward the simple subpopulation schema based on sharing schema to reduce the computation complexity efficiently. In 2001, Dr Liu[6] brought forward two kinds of new MGA (balanced space method and local sharing method) by which the optimization of uneven peak function can be solved efficiently. Dr Yang also discusses the optimization of multi-modal function through improving on the Immune algorithm [7].

Currently, most of the multi-modal algorithms such as GA and Immune Algorithm not only have the shortcoming of missing peak value, needing much of population but also need some rigorous prior knowledge such as the number of peaks and the peak distance more or less. These factors embarrass the application of optimization algorithm. In this article a new multi-modal optimization algorithm (SOMA) is brought forward for actual application of multi-modal problem from a new point of view. Through studying and chasing among the worms this algorithm can find all peaks of multi-modal function. This algorithm has the character of a simple calculation, strong convergence and high precision. Especially there is no need for any prior knowledge in optimization.

* This project was supported by the National Natural Science Foundation of China (70572045).

2. The self-organizing worm algorithm

The main idea of SOWA can be described as follows: disperse some worm equably in the domain; define the simple rule for each worm that climbs toward the high point; if a worm has already been at the high point (locations of all his neighbor worms are below it) it will stop creeping and its neighbor worms will creep toward it. Through the ceaseless creeping of worms in the domain, all worms will concentrate on the peaks of the multi-modal function.

In SOWA, each worm is named as a unit and the neighbor worms as neighbor units.

2.1 Description of SOWA

First, let the dimensionality of functions to be optimized be l. Select m units $k_i(i = 1, 2, \cdots, m)$ (a unit is a possible solution of the problem; the selection method can be assured according to the dimensionality and the characteristic of the function) in domain to build up the initial group. Select p neighbors (the number of neighbors can be assured by the dimensionality of functions, it can be defined as p=2l). The neighbors of unit k_i can be marked as neighbor $_{ij}(j$=1, 2, $\cdots, p)$. Next compute the location parameter $x_{ei}(e$=1, 2, $\cdots, l)$and the fitness value Fitness (k_i) which can be defined by the object function.

Second, define the motion rules of units $k_i(i$=1,2, $\cdots, m)$: compare the Fitness (k_i) of k_i with its neighbors neighbor $_{ij}(j$=1, 2, $\cdots, p)$. If Fitness (k_i) is more than all its neighbors neighbor$_{ij}(j$=1, $\cdots, p)$ the unit k_i stops moving and sets the motion parameter. Forward (k_i)=0; or else sets Forward (k_i)=1 and moves toward the nearest neighbor whose fitness is more than units k_i . The new location of unit k_i will be the golden section between k_i and this neighbor. Compute the new Fitness (k_i) of k_i, return to the former location if the new value is less than the former value. A cycle is completed when all units in the domain have performed once according to the rules.

All units in-group change their locations continuously until stop conditions have been matched.

2.2 The flow diagram of SOWA

procedure worm
begin
Initialize();
For(i=1;i<n;i++)
Move() ;
Output();
End

2.1.1 Initialization (initialize ())

Select m units $k_i(i$=1,2,$m)$ in domain (x_{e0}, x_{en}) to build up the initial group. The location parameter can be calculated as follows: divide the domain of every dimensionality into$s$parts, the units will be placed at all crossings of these divisions. The number of units can be calculated as shown in Eq.(1). The location parameters $x_{ei}(e$=1,2,$\cdots, l)$ and the Fitness (k_i) of unit k_i can be assured as Eq.(2) and Eq.(3).

$$m = (s+1)^l \tag{1}$$

$$\begin{gathered} x_{ei} = x_{e0} + (\frac{x_{en} - x_{e0}}{s})(q_{ei} - 1) \\ i = 1, 2, \cdots, m, e = 1, 2, \cdots, l, q_{ei} = 1, 2, \cdots, s \end{gathered} \tag{2}$$

$$\begin{gathered} Fitness(k_i) = f(x_{1i}, x_{2i} \ldots x_{li}) \\ i = 1, 2, \cdots, m \end{gathered} \tag{3}$$

For unit k_i,$x_{ei}(e$=1,2,$\cdots$,l) denotes the coordinate of a certain dimensionality, q_{ei} denotes the location of a certain dimensionality coordinate in the domain of this dimensionality.

2.1.2 Motion rules (Move ())

If Fitness (k_i) is more than all its neighbors (neighbor$_{ij}(j$=1,$\cdots, p))$, the moving of unit k_i is stopped and set the motion parameter Forward (k_i)=0; or else set Forward (k_i)=1 and moves toward the nearest neighbor whose fitness is more than units k_i. Set the location parameters of this neighbor as minx $_{ei}(e$=1,2,$\cdots, l)$ whose step length can be calculated as Eq.(4). Compute the new Fitness (x_i) of k_i and return to the former location if the new value is less than the former value.

$$\begin{gathered} Pace_i = \sqrt{\sum_{e=1}^{l} (x_{ei} - \min x_{ei})^2} * 0.618 \\ e = 1, 2, \cdots, l \end{gathered}$$

2.1.3 Output rules (Output ())

After n^{th} cycle units k_i(Forward (k_i)=0) are on peak points. The locations and the fitness values of k_i will