Neurocomputing 137 (2014) 261–267

Contents lists available at ScienceDirect

Neurocomputing

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

A particle swarm optimization using local stochastic search and enhancing diversity for continuous optimization

Jianli Ding [a], Jin Liu [a,b,*], Kaushik Roy Chowdhury [c], Wensheng Zhang [d], Qiping Hu [e], Jeff Lei [f]

[a] State Key Laboratory of Software Engineering, Computer School, Wuhan University, Wuhan 430072, China
[b] School of Computer Science & Technology, Nanjing University of Posts and Telecommunications, Nanjing 210046, China
[c] Electrical and Computer Engineering Department, Northeastern University, Boston, MA 02115, USA
[d] State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Science, Beijing 100190, China
[e] International School of Software, Wuhan University, Wuhan 430072, China
[f] Department of Computer Science and Engineering, University of Texas at Arlington, Arlington, TX 76019, USA

ARTICLE INFO

Article history:
Received 7 January 2013
Received in revised form
5 March 2013
Accepted 27 March 2013
Available online 27 February 2014

Keywords:
Particle swarm optimization
Local stochastic search
Diversity
Continuous optimization problems

ABSTRACT

The particle swarm optimizer (PSO) is a swarm intelligence based on heuristic optimization technique that can be applied to a wide range of problems. After analyzing the dynamics of traditional PSO, this paper presents a new PSO variant on the basis of local stochastic search strategy (LSSPSO) for performance improvement. This is encouraged by a social phenomenon that everyone wants to first exceed the nearest superior and then all superior. Specifically, LSSPSO employs a local stochastic search to adjust inertia weight in terms of keeping a balance between the diversity and the convergence speed, aiming to improve the performance of traditional PSO. Experiments conducted on unimodal and multimodal test functions demonstrate the effectiveness of LSSPSO in solving multiple benchmark problems as compared to several other PSO variants.

1. Introduction

Particle swarm optimization (PSO) is a population based, intelligent, guided, and stochastic search technique that was developed by Eberhart and Kennedy [1,2] based upon the concept of swarming. The PSO uses a simple mechanism that imitates their swarm behaviors to guide the particles to search for globally optimal solutions. Similar to other evolutionary computation technique [3–6], it is also a population-based iterative algorithm. Because of its simplicity of implementation and capability to quickly converge to a reasonably good solution [3,4], the PSO has been successfully applied in solving many real-world optimization problems [5–7].

However, the PSO has difficulties in keeping the balance between exploration and exploitation when the application environment is dynamic [8–10]. In other words, the PSO cannot adapt to the changing environment and converge to an optimum in an early period of iteration [11,12]. Another main drawback of the PSO is that it may get stuck at a local-optimal solution region [13].

Therefore how to accelerate the convergence speed and how to avoid the local optimal solution are two important issues in the PSO research. In general, research on PSO algorithm development can be classified in three categories [3]:

(1) Parameter selection for the particle swarm: when implementing the particle swarm algorithm, parameter selections must be taken into account to facilitate the convergence and prevent an "explosion" of the swarm. These parameter selections include limiting the maximum velocity, selecting acceleration constants, the constriction factor, or the inertia constant. Inertia weight is the most important parameter of PSO, a linearly varying inertia weight [14], the time-varying acceleration coefficients [15], a fuzzy adaptive inertia weight [16] and a random inertia weight [17] are used to improve the search performance of PSO. The restriction factor is another parameter of PSO. Eberhart and Shi [18,19] pointed out that the factor for analyzing the convergence behavior is equivalent to the inertia weight. By evaluating the population distribution and particle fitness, Zhan et al. proposed the so-called adaptive PSO (APSO) [3] which utilizes an evolutionary state estimation (ESE) technique to ascertain one out of four defined evolutionary states (i.e., exploration, exploitation, convergence and jumping out) in each iteration.

* Corresponding author at: State Key Laboratory of Software Engineering, Computer School, Wuhan University, Wuhan 430072, China.
Tel.: +86 27 6877 6052.
E-mail addresses: dingjianli@whu.edu.cn (J. Ding), mailjinliu@yahoo.com (J. Liu), krc@ece.neu.edu (K.R. Chowdhury), wensheng.zhang@ia.ac.cn (W. Zhang), huqp@whu.edu.cn (Q. Hu), ylei@cse.uta.edu (J. Lei).

http://dx.doi.org/10.1016/j.neucom.2013.03.075
0925-2312

(2) Hybrid versions of the particle swarm: the second category of PSO research is to investigate different learning strategies on exemplar (pBest and gBest) selection for the particle to quickly converge to near-optimum (if not optimum) solutions. A natural evolution of the population based search algorithms like that of PSO can be achieved by incorporating the methods that have already been examined successfully for solving complex problems. Some recent research indicates that the performance of PSO (e.g., convergence rate, solution quality) could be much improved via a model fusion concept, that is, integrating PSO with other search techniques, such as evolutionary operators: selection, crossover, and mutation [20,21]; or evolutionary algorithms: genetic algorithm (GA), simulated annealing (SA), memetic algorithm (MA), and cellular automata (CA) [22–24]. The main goal, as we see is to harness the strong points of the algorithms in order to keep a balance between the exploration and exploitation factors thereby preventing the stagnation of population and preventing premature convergence.

(3) Topology of the particle swarm: in order to improve the performance of PSO, different types of topologies have been advanced. In [25,26], Kennedy analyzed the effects of neighborhood topology on PSO, and suggested four different neighborhood topologies (circle topology, wheel topology, star topology, and random topology [28]). The experiment results show that PSO with a small neighborhood may perform better on complex problems, while PSO with a large neighborhood may perform better on simple problems. In [27], Suganthan introduced a neighborhood operator which gradually increases the neighborhood size of a particle until it covers all particles in a swarm. In [28], Hu and Eberhart updated the neighborhood of each particle by dynamically selecting m particles that are the nearest to the current particle. In [29], Mendes and Kennedy proposed a fully informed PSO algorithm (FIPS), in which the neighbors of each particle, instead of *pbest* and *gbest*, are used to update the velocity. Based on the FIPS, in [30], Mohais et al. proposed random and dynamic neighborhoods by re-structuring neighborhoods in terms of a diversity-preserving measure. In [31], Peram et al. presented a modified PSO called fitness-distance-ration-based PSO (FDR-PSO), which uses a new velocity updating method. In [32], Yano et al. proposed a hybrid PSO algorithm with a neighborhood search. When the current position of a particle is better than its previous best position, the algorithm will search 2^D-1 points in the neighborhood of the current point (D is the dimension size). However, the algorithm converges very slowly because the neighborhood search is time consuming.

In terms of this, we propose a new PSO variant algorithm using local stochastic search strategy (LSSPSO), inspired by a social phenomenon that each individual first wants to stand out among its neighbors and then stand out among the whole community. With the proposed approach, each particle has its own inertia weight and acceleration coefficients. Even if in the same iteration, those parameters may tend to differ for various particles. Also they are time-varying over the iterations. The proposed LSSPSO algorithm regulates inertia weight in such a way that it does not reduce the diversity rapidly and at the same time prevents early convergence to local optima. More specifically, LSSPSO introduces a new guidance to the particle trajectory model to increase the diversity of the particle swarm without decreasing the convergence speed. In addition, the purpose of the proposed parameter automation strategies is to speed up the convergence speed and avoid the local optimal solution, in order to enable the LSSPSO to perform a global search over the search space with faster convergence speed. Experimental results demonstrate the effectiveness of LSSPSO in solving multiple benchmark problems.

The remainder of this paper is organized as follows: Section 2 briefly presents some background material of particle swarm optimization and describes some variants of PSO. The proposed local stochastic search particle swarm optimization is described in Section 3. Section 4 presents the experimental results and show the effectiveness and efficiency of the proposed LSSPSO algorithm through comparing to several other well-known PSO variants. Finally, Section 5 contains some conclusions of this study.

2. Overview of the related works

2.1. Classical PSO

In PSO, each solution vector is known as a particle and several such particles collectively form a swarm. Each member in the swarm adapts its search patterns by learning from its own experience as well as other particles. The particle has a tendency to move towards a better search area with a definite velocity determined by the information collected by the particle over the course of the search process.

While searching in a D-dimensional hyperspace, each particle i has a position vector $X_i=[x_i^1,x_i^2,\cdots,x_i^D]$ and a velocity vector $V_i=[v_i^1,v_i^2,\cdots,v_i^D]$ to indicate its current state where i is a positive integer indexing the particle in the swarm and D is the dimension of the problem under study. Moreover, particle i will keep its personal historical best position vector $pbest_i=[pbest_i^1,pbest_i^2,\cdots,pbest_i^D]$. The best position of all the particles in the current step is $gbest=[gbest^1,gbest^2,\cdots,gbest^D]$. The vectors x_i and v_i are initialized randomly and are updated by using the following formulae:

$$v_i^d=v_i^d+c_1rand_1(pbest_i^d-x_i^d)+c_2rand_2(gbest^d-x_i^d) \tag{1}$$

$$x_i^d=x_i^d+v_i^d \tag{2}$$

where d denotes the dimension in the solution space, corresponding to which the velocity is updated. Coefficients c_1 and c_2 are the acceleration parameters reflecting the weighting of stochastic acceleration terms that pull each particle towards *pbest* and *gbest* positions, respectively. These are usually set to 2. Sometimes they may be adaptively controlled according to the evolutionary states. $rand_1$ and $rand_2$ represent separately two randomly generated numbers in the range [0, 1]. A particle's velocity is clamped to a maximum magnitude v_{max}. That it is to say, if $|v_i^d|$ exceeds a positive constant value specified by the user, then the velocity of that dimension is set to $sign(v_i^d)v_{max}^d$, where $sign(x)$ is the triple-valued signum function [33].

It is noteworthy that Eq. (1) presents the primitive form of velocity updating formula. How to control or adjust the flying velocity, an inertia weight or a constriction factor was introduced by Shi and Eberhart [34], and Clerc and Kennedy [4], respectively. Using the inertia factor ω, the new velocity updating formula becomes

$$v_i^d=\omega v_i^d+c_1rand_1(pbest_i^d-x_i^d)+c_2rand_2(gbest^d-x_i^d) \tag{3}$$

Using the constriction factor χ, the new velocity updating formula becomes,

$$v_i^d=\chi[v_i^d+c_1rand_1(pbest_i^d-x_i^d)+c_2rand_2(gbest^d-x_i^d)] \tag{4}$$

where $\chi=\frac{2}{\left|2-\phi-\sqrt{\phi^2-4\phi}\right|}$ and $\phi=c_1+c_2$.

2.2. Some variants of PSO

In population-based optimization methods, considerably high diversity is necessary during the early part of the search to allow