Expert Systems With Applications 92 (2018) 256–275

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Hybrid non-parametric particle swarm optimization and its stability analysis

Zhao-Guang Liu[a,*], Xiu-Hua Ji[a], Yun-Xia Liu[b]

[a] School of Computer Science and Technology, Shandong University of Finance and Economics, Jinan, China
[b] School of Information Science and Engineering, University of Jinan, Jinan, China

ARTICLE INFO

Article history:
Received 18 May 2017
Revised 8 August 2017
Accepted 9 September 2017
Available online 22 September 2017

Keywords:
Particle swarm optimization
Hybrid algorithm
Parameter selection
Stability analysis

ABSTRACT

As a population-based random search optimization technique, particle swarm optimization (PSO) has become an important branch of swarm intelligence (SI). The tuning of parameters in PSO has attracted the attention of many researchers. This study proposes an alternative technology called hybrid non-parametric PSO (HNPPSO) algorithm. Other SI operations, including a multi-crossover operation, a vertical crossover, and an exemplar-based learning strategy, are combined with the proposed algorithm to balance the global and local search capabilities. The first- and second-order stability analyses conducted for the present study showed that the particle positions are expected to converge at a fixed point in the search space and that the variance of the particle positions converges to zero. In the experiments, the proposed algorithm was compared with 10 other advanced PSO techniques using 40 widely used benchmark functions. The experimental results indicated that the proposed algorithm yields better solution accuracy and convergence speed than the other PSO techniques. The proposed algorithm significantly outperformed the other PSO approaches in terms of convergence speed.

1. Introduction

Inspired by the swarming behavior of flocking birds, Kennedy and Eberhart (1995) introduced the particle swarm optimization (PSO) algorithm. Each particle in PSO represents a possible solution for an optimization problem in a D-dimensional space and has a velocity vector $V_i = [v_i^1, v_i^2, \ldots, v_i^D]$ and a position vector $X_i = [x_i^1, x_i^2, \ldots, x_i^D]$. Each particle flies within the search space and is attracted by its own previous best position (*pbest*) and by the swarm's global best position (*gbest*). The velocity and the position for the *i*th ($i = 1, 2, \ldots, N$) particle during the *t*th ($t = 1, 2, \ldots, T$) iteration are updated as follows:

$$v_i^d(t+1) = wv_i^d(t) + c_1r_1\left(p_i^d(t) - x_i^d(t)\right) + c_2r_2(g^d(t) - x_i^d(t)) \quad (1)$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \quad (2)$$

where p_i is $pbest_i$; g is *gbest;* N is the population size; d ($d=1, 2, \ldots, D$) denotes the dimensions of the solution space; T is a predefined maximum number of iterations; c_1 and c_2 are the acceleration coefficients indicating the influence of the particle's *pbest* and *gbest*, respectively; and r_1 and r_2 are two uniformly distributed random numbers within the range of [0, 1]. Parameter w is the inertia weight used to balance the global and local search capabilities (Kennedy, 1999).

The PSO algorithm has attracted the attention of many researchers during the past few decades because of its simple concept and easy implementation. The algorithm has also been successfully applied to a wide range of applications (Lien & Cheng, 2012; Mandloi & Bhatia, 2016; Wang & Yeh, 2014). Although PSO has an effective search capability when solving a unimodal problem, it may get trapped at the local minimum when applied to more complex problems, which, consequently, may lead to a premature convergence (Chen et al., 2013). Many methods have been proposed to improve the PSO performance. These methods can be roughly classified into the following categories: (i) learning strategies, (ii) multi-swarm schemes, (iii) neighborhood topologies, (iv) combinations with other swarm intelligence (SI) algorithms, and (v) parameter selection. In (i), some PSO variants are inspired by various learning mechanisms (Cheng & Jin, 2015; Lim & Isa, 2014a; Tanweer et al., 2015) to update the particle velocity and position. In (ii), a swarm is divided into several layers known as sub-swarms (Gülcü & Kodaz, 2015; Tanweer et al., 2016; Wang et al., 2014). Sub-swarms can explore the different regions of the solution space with different algorithms. Meanwhile, in (iii), the neighborhood topologies of the particles are researched

* Corresponding author.
E-mail addresses: liuzhg@sdufe.edu.cn (Z.-G. Liu), jane@sdu.edu.cn (X.-H. Ji), ise_liuyx@ujn.edu.cn (Y.-X. Liu).

https://doi.org/10.1016/j.eswa.2017.09.012
0957-4174/

(Fang et al., 2016; Kennedy, 1999; Marinakis & Marinaki, 2013), including fixed neighborhood topologies and variable neighborhood searches. The last two methods, namely, combinations with other SI algorithms and parameter selection, are discussed in Section 2.

Convergence to a point for the standard PSO algorithm is usually analyzed to determine the coefficients' boundaries, where the generated solutions by the algorithm do not diverge. These boundaries are known as convergence boundaries. The convergence and stability of standard PSO have been investigated, including the deterministic model stability analysis (Clerc & Kennedy, 2002), first-order stability analysis (Clerc, 1999; Cleghorn & Engelbrecht, 2014; Bergh & Engelbrecht, 2006), and second-order stability analysis (Bonyadi & Michalewicz, 2016; García-Gonzalo & Fernández-Martínez, 2014; Liu, 2014; Poli, 2009).

According to some researchers' study (Jiang et al., 2007; Poli, 2009), first-order stability is not enough to ensure convergence; the second-order stability condition must also be satisfied to ensure the convergence of the variance or the standard deviation.

The remainder of the paper is organized as follows. The backgrounds of the hybrid PSO variants, exemplar-based learning strategy, and parameter selection are presented in Section 2; the proposed hybrid non-parametric particle swarm optimization (HNPPSO) technique is discussed in Section 3; a stability analysis of the proposed algorithm is discussed in Section 4; the benchmark functions used to test HNPPSO and compare it with 10 other start-of-the-art PSO variants are presented in Section 5; and, finally, the concluding remarks are provided in Section 6.

2. Background

The following three related technologies will be discussed in this section: hybrid PSO variants, exemplar-based learning strategy, and parameter selection in PSO.

2.1. Hybrid PSO variants

Combining PSO with other SI algorithms is a commonly used technique to overcome the weaknesses of PSO. Such hybrid algorithms include the genetic algorithm (GA) (Juang, 2004), differential evolution (DE) (Lin et al., 2016; Nwankwor et al., 2013), chemical reaction optimization (Nguyen et al., 2014), artificial bee colony (Li, Zhan, Lin, Zhang, & Luo, 2015; Li, Wang, Yan, & Li, 2015), and ant colony optimization (Mandloi & Bhatia, 2016; Shelokar et al., 2007).

Some PSO variants adopted a two-swarm scheme (Lin et al., 2016; Shelokar et al., 2007) to combine with other SI algorithms. The evolution strategy of the particles in one swarm in such variants is applied using PSO. The SI algorithm is applied to a second swarm. Kao (2008) proposed a hybrid between a GA and a PSO algorithm. In this proposal, GA operations (i.e., crossover and mutation) are applied to the top 2N individuals with better fitness values after generating a population of size 4N, whereas PSO operations (i.e., velocity and position updates) are applied to the 2N individuals with the worst fitness values. In another hybrid GA and PSO variant proposed by Juang (2004), three major operations precede a particle swarm in a cascade-like manner: *enhancement, crossover*, and *mutation*. The *enhancement* operation is implemented using PSO, whereas the other two operations are implemented using GA. Lin et al. (2016) proposed a hybrid between PSO and DE. DE is used to provide the necessary momentum for the particles to roam across the search space and escape from the local optimum when a swarm settles into a stagnation state.

Some PSO variants were combined with certain operations of other SI algorithms to improve the PSO performance (Mahmoodabadi et al., 2014; Soleimani, & Kannan, 2015; Wu et al., 2014; Zhang & Xie, 2003). For instance, in high-exploration PSO (HEPSO) (Mahmoodabadi et al., 2014), one operation was inspired by the multi-crossover mechanism of GA. Another operation uses a bee colony mechanism to update the particle positions. Soleimani and Kannan (2015) also invoked the crossover operation in GA in a proposed hybrid variant. The PSO and DE operations in the hybrid algorithm proposed by Zhang and Xie (2003) are alternately performed (i.e., the velocity and position updates are conducted by PSO during odd generations and a mutation operation for *pbest* is conducted during even generations).

2.2. Exemplar-based learning strategy

Some PSO variants adopted the concept of an exemplar to direct the flying of particles, including the comprehensive learning particle swarm optimization (CLPSO) (Liang et al., 2006), competitive and cooperative particle swarm optimization (CCPSO) (Li & Zhan et al., 2015; Li & Wang et al., 2015), orthogonal learning particle swarm optimization (OLPSO) (Zhan et al., 2011), and scatter learning PSO (SLPSO) (Ren et al., 2014). The velocity in these learning strategies is updated through the following equation:

$$v_i^d(t+1) = wv_i^d(t) + c \cdot r^d \left(exemplar_i^d(t) - x_i^d(t) \right) \tag{3}$$

where $exemplar_i$ is the exemplar of the ith particle on the dth dimension and r is a uniformly distributed random number within the range of [0, 1].

How to choose an exemplar for the ith particle is the main research topic in these articles. With CLPSO (Liang et al., 2006), two particles are randomly chosen out of the population. The fitness values of the two particles' *pbest*s are compared, and the one with the better fitness is selected as the exemplar. CCPSO (Li & Zhan et al., 2015; Li & Wang et al., 2015) is implemented by using a sharing device called "blackboard." In each iteration, all the particles post their *pbest*s to the blackboard. For the dth dimension of the ith particle, the particle communicates with all the particles through the blackboard and randomly selects some particles to compete against. The winner with the best *pbest* fitness becomes the exemplar. With OLPSO (Zhan et al., 2011), the exemplar comes from *pbest* or *gbest* resulting from the construction of the experimental orthogonal design. Meanwhile, with SLPSO (Ren et al., 2014), an exemplar pool (EP) composed of a certain number of relatively high-quality solutions scattered in the solution space is constructed and requires particles to select their exemplars from the EP using a roulette wheel rule.

2.3. PSO parameters

The PSO described in Eqs. (1) and (2) includes three parameters, namely, w, c_1, and c_2. The inertia weight w introduced by Shi and Eberhart (1998a) is used to control the balance between the global and the local exploration capabilities. A larger inertia weight w facilitates a global exploration, whereas a smaller one tends to facilitate a local exploration for fine-tuning the current search area. A suitable selection of the inertia weight w can provide a balance between the global and the local exploration capabilities, thereby requiring fewer iterations on average to find the optimum solution (Shi & Eberhart, 1998b). Various methods for adjusting w have been proposed to improve the PSO performance. These methods can be classified as linear (Eberhart & Shi, 2000; Zhou et al., 2011), nonlinear (Chatterjee & Siarry, 2006), random (Clerc, 1999; Eberhart & Shi, 2001; Khan et al., 2016), and adaptive (Ardizzon et al., 2015; Taherkhani & Safabakhsh, 2016; Wang & Yang, 2016; Zhan et al., 2009).

The acceleration coefficients c_1 and c_2 represent the weighting of the stochastic acceleration terms: a cognitive component (*pbest*) and a social component (*gbest*). Therefore, the proper control of these two components is very important in finding the optimum solution in an accurate and efficient manner. An increase