Expert Systems With Applications 62 (2016) 317–332

Contents lists available at ScienceDirect

Expert Systems With Applications

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

A differential-based harmony search algorithm for the optimization of continuous problems

Hosein Abedinpourshotorban[a,b], Shafaatunnur Hasan[a,b], Siti Mariyam Shamsuddin[a,*,b], Nur Fatimah As'Sahra[b]

[a] *UTM Big Data Centre, Ibnu Sina Institute for Scientific and Industrial Research, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia*
[b] *Faculty of Computing, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia*

ARTICLE INFO

Article history:
Received 3 November 2015
Revised 4 February 2016
Accepted 7 May 2016
Available online 10 May 2016

Keywords:
Harmony search algorithm
Continuous optimization
Evolutionary optimization
Differential evolution
Meta-heuristics

ABSTRACT

The performance of the Harmony Search (HS) algorithm is highly dependent on the parameter settings and the initialization of the Harmony Memory (HM). To address these issues, this paper presents a new variant of the HS algorithm, which is called the DH/best algorithm, for the optimization of globally continuous problems. The proposed DH/best algorithm introduces a new improvisation method that differs from the conventional HS in two respects. First, the random initialization of the HM is replaced with a new method that effectively initializes the harmonies and reduces randomness. Second, the conventional pitch adjustment method is replaced by a new pitch adjustment method that is inspired by a Differential Evolution (DE) mutation strategy known as *DE/best/1*. Two sets of experiments are performed to evaluate the proposed algorithm. In the first experiment, the DH/best algorithm is compared with other variants of HS based on 12 optimization functions. In the second experiment, the complete CEC2014 problem set is used to compare the performance of the DH/best algorithm with six well-known optimization algorithms from different families. The experimental results demonstrate the superiority of the proposed algorithm in convergence, precision, and robustness.

1. Introduction

Optimization refers to the process of selecting the best solution from the set of all possible solutions to maximize or minimize the cost of the problem (Moh'd Alia & Mandava, 2011). Optimization problems can be categorized into discrete or continuous groups based on the solution set (Velho, Carvalho, Gomes, & de Figueiredo, 2011). An additional category is based on the properties of the objective function, such as unimodal or multimodal.

Therefore, various optimization algorithms are required to tackle different problems. There are two types of optimization algorithms: exact and approximate (Stützle, 1999). Exact algorithms are guaranteed to find the best solution within a certain period of time (Weise, 2009). However, real world problems are mostly NP-hard, and finding the solutions for this type of problem using exact algorithms consumes exponential amounts of time (Johnson, 1985; Michael & David, 1979). Thus, approximate algorithms have been applied recently to find near-optimal solutions to NP-hard problems in reasonable amounts of time.

Meta-heuristics are approximate algorithms that are able to find satisfactory solutions for optimization problems in reasonable amounts of time (Blum & Roli, 2003, 2008). Meta-heuristics are also used to address a major drawback of approximate local search algorithms, which is finding local minima instead of global minima.

Differential Evolution (DE) (Price, Storn, & Lampinen, 2006; Storn & Price, 1995, 1997) emerged in the late 1990 s and is one of the most competitive metaheuristic algorithms. The DE algorithm is somewhat similar to the Genetic Algorithm (GA), but the solutions consist of real values instead of binary values and generally converge faster than the GA (Hegerty, Hung, & Kasprak, 2009). The performance of DE greatly depends on the parameter settings (Islam, Das, Ghosh, Roy, & Suganthan, 2012). Many variants of DE have been proposed to address different problems, but DE still faces several difficulties in optimizing some types of functions as has been pointed out in several recent publications (Hansen & Kern, 2004; Ronkkonen, Kukkonen, & Price, 2005). However, due to the optimization power of the DE algorithm, it is commonly applied for the optimization of real world problems, such as optimizing compressor supply systems (Hancox & Derksen, 2005),

* Corresponding author. Tel.: 6075531993; Fax: 6075565044.
E-mail addresses: h_abedinpour@gmail.com (H. Abedinpourshotorban), shafaatunnur@utm.my (S. Hasan), mariyam@utm.my (S.M. Shamsuddin), fatim.assahra@gmail.com (N.F. As'Sahra).

http://dx.doi.org/10.1016/j.eswa.2016.05.013
0957-4174/

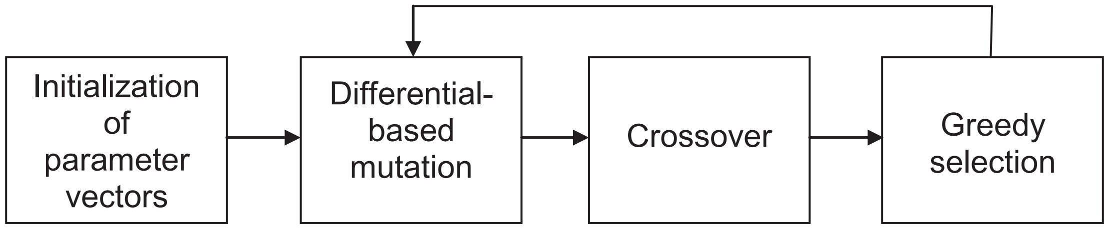

Fig. 1. General process of DE.

determining earthquake hypocenters (Růžek & Kvasnička, 2005) and for 3D medical image registration (Salomon, Perrin, Heitz, & Armspach, 2005).

Another recent and well-known meta-heuristic algorithm is Harmony Search (HS), which was proposed by Geem, Kim, and Loganathan (2001). HS is inspired by the way that musicians experiment and change the pitches of their instruments to improvise better harmonies. The HS algorithm has been applied to many optimization problems, such as the optimization of heat exchangers, telecommunications, vehicle routing, pipe network design, and so on (Cobos, Estupiñán, & Pérez, 2011; Geem, Kim, & Loganathan, 2002; Geem, Lee, & Park, 2005; Manjarres et al., 2013; Omran & Mahdavi, 2008; Pan, Suganthan, Tasgetiren, & Liang, 2010; Wang & Yan, 2013; Xiang, An, Li, He, & Zhang, 2014).

Although HS has achieved significant success, several shortcomings prevent it from rapidly converging toward global minima. HS generally has inadequate local search power due to its reliance on the parameter settings, which greatly affects its performance. The Improved Harmony Search (IHS) (Mahdavi, Fesanghary, & Damangir, 2007) was proposed to address the local search issue of HS by dynamically adjusting the Pitch Adjustment Rate (PAR) and Bandwidth (bw) when the algorithm is run. However, IHS has a high demand on the parameter settings prior to starting the optimization process. To dynamically adjust the HS control parameters with respect to the evolution process and the search space of optimization problems, Chen et al. (Chen, Pan, & Li, 2012) introduced a new variant of HS called NDHS. Although NDHS outperforms conventional HS and IHS, some of the parameters must be set before the search process begins. To eliminate the labor that is associated with setting the parameters, a new parameter-less variant of HS, called GDHS, was proposed by Khalili, Kharrat, Salahshoor, and Sefat (2014) , which outperforms the existing variants of HS.

Although many variants of HS have been proposed, the demand for improvement in evolutionary algorithms is increasing as real world problems become increasingly complicated. Motivated by the facts that the bw parameter of the HS algorithm is problem-dependent and significantly influences the performance of the algorithm and that most of the existing methods address this issue by dynamically changing the bw based on the number of harmony improvisations instead of considering the problem's surface, this paper proposes a new variant of HS, called the DH/best algorithm, to eliminate the need for bw and to improve the accuracy and robustness of HS.

2. Methodology

In this section, we study the DE and HS algorithms in detail. Later in the paper (Section 4), we propose a new hybrid algorithm by combining these two algorithms.

2.1. Differential Evolution

The DE algorithm is a population-based global optimizer that outperforms many optimization algorithms in terms of robustness and convergence speed. Many versions of DE have been developed. The original version of DE is known as *DE/rand/l/bin* or "classic DE" (Storn & Price, 1997). The DE variant that is used in this paper is called *DE/best/1*; it differs from the classical version of DE only in terms of its mutation strategy. Experiments by Geem, 2010; Mezura-Montes, Velázquez-Reyes, and Coello Coello (2006) on various types of optimization functions demonstrated that the *DE/best/1* scheme is the most competitive DE scheme regardless of the characteristics of the problem to be solved. The general process of the DE algorithm is illustrated in Fig. 1.

Step 1: Initialization of parameter vectors

A random population of parameter vectors is initialized in this step. The number of parameters of each vector is equal to the number of problem parameters, and each parameter of the vector corresponds to one problem variable. In addition, the value of each parameter is allocated randomly from the parameter range. A DE parameter vector is defined as:

$$\overrightarrow{X_{i,G}} = [x_{1,G}, x_{2,G}, x_{3,G}, \ldots,\ x_{D,G}] \tag{1}$$

where D is the number of parameters, which will not change during the optimization, and G indicates the current generation and increases generation by generation.

Step 2: Differential-based mutation

The new parameter vectors are generated by adding the weighted difference between two randomly selected population vectors to the best vector (fittest). The mutant vector is calculated by Eq. (2):

$$\overrightarrow{v_{i,G+1}} = \overrightarrow{X_{best,G}} + F.\left(\overrightarrow{X_{r1,G}} - \overrightarrow{X_{r2,G}}\right) \tag{2}$$

where *best* is the index of the fittest vector in the population, and "r_1" and "r_2" are the indexes of randomly selected vectors from the population and are different from the index of the best vector.

Step 3: Crossover

To increase the diversity of the population, DE mixes the mutated vector's parameters with the parameters of another predetermined vector, which is called the target vector. The new vector is called the trial vector. The generation of the trial vector is formulated in Eq. (3):

$$u_{j\ i,\ G+1} = \begin{cases} v_{j\ i,\ G+1} \ if\ \ (randb(j) \le CR)\ \ or\ \ j = rnbr(i) \\ x_{j\ i,\ G} \ if\ \ (randb(j) > CR) and\ \ j \ne rnbr(i) \end{cases}$$
$$j = 1, 2, \ldots, D \tag{3}$$

where $randb(j)$ is the jth evaluation of a random function, which generates uniform random numbers between (0 and 1), CR is the crossover constant (between 0 and 1), which is determined by the user, and $rnbr(i)$ is a randomly selected number $\epsilon\ 1, 2, \ldots, D$ that ensures that $u_{i,\ G+1}$ obtains at least one parameter from $v_{i,\ G+1}$.