Swarm and Evolutionary Computation xxx (xxxx) xxx–xxx

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

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

Global-best brain storm optimization algorithm

Mohammed El-Abd

Electrical and Computer Engineering Department, American University of Kuwait, P.O. Box 3323, Safat 13034, Kuwait

ARTICLE INFO

Keywords:
Brain storm optimization
Global-best
Per-variable updates
Re-initialization
Fitness-based grouping
Unconstrained optimization

ABSTRACT

Brain storm optimization (BSO) is a population-based metaheuristic algorithm that was recently developed to mimic the brainstorming process in humans. It has been successfully applied to many real-world engineering applications involving non-linear continuous optimization. In this work, we propose improving the performance of BSO by introducing a global-best version combined with per-variable updates and fitness-based grouping. In addition, the proposed algorithm incorporates a re-initialization scheme that is triggered by the current state of the population. The introduced Global-best BSO (GBSO) is compared against other BSO variants on a wide range of benchmark functions. Comparisons are based on final solutions and convergence characteristics. In addition, GBSO is compared against global-best versions of other meta-heuristics on recent benchmark libraries. Results prove that the proposed GBSO outperform previous BSO variants on a wide range of classical functions and different problem sizes. Moreover, GBSO outperforms other global-best meta-heuristic algorithms on the well-known CEC05 and CEC14 benchmarks.

1. Introduction

One class of algorithms used to solve non-linear continuous and/or discrete optimization problems is Population-based algorithms. Population-based algorithms maintain a population of individuals (solutions) and update them over a number of iterations (generations) until some stopping criterion is met. Population-based algorithms could be further categorized based on the inspiration behind their population update mechanism. The first category is evolutionary algorithms, in which the update process is inspired by the biological evolution process. These algorithms include Genetic Algorithms (GAs), Genetic Programming (GP), Evolutionary Strategies (ES), and Evolutionary Programming (EP). The second category includes swarm intelligence algorithms, in which the update process is inspired by some behavior of some living organism. A number of swarm intelligence algorithms are referred to as foraging algorithms as they mimic the foraging behavior of animals and/or insects. Examples of foraging algorithms include Particle Swarm Optimization (PSO) [1,2], Ant Colony Optimization [3], Artificial Bee Colony [4], and many more. Other swarm intelligence algorithms are inspired by different kinds of behaviors including for example the egg laying behavior of cuckoos in Cuckoo Search (CS) [5] and the echolocation behavior of bats in the Bat Algorithm (BA) [6].

The brain storm optimization (BSO) algorithm [7,8] is a population-based algorithm proposed to mimic brainstorming sessions held by humans. A typical brainstorming session involves gathering a group of experts having different backgrounds, expertise, and abilities in order to develop a solution for a problem at hand. Following such a process helps in successfully solving the tackled problem. The first version of the developed BSO algorithm had a number of disadvantages including the need to provide the number of clusters before hand, the computational complexity of the clustering stage, the lack of a re-initialization step, and the fixed schedule for updating the step size. Some of these disadvantages have already been addressed in the literature by either improving the clustering step [9–11], provide a better update method [9,12,13], or introduce a re-initialization mechanism [14–16]. However, to the best of our knowledge, a global-best version of BSO has not been proposed before.

In this paper we propose multiple modifications to improve the performance of BSO. These modifications include adopting a fitness-based grouping mechanism, using the global-best idea information for updating the population, and applying the update scheme on every problem variable separately. The proposed Global-best BSO (GBSO) is compared against three recent variants of BSO using a suite of 20 well-known benchmark functions. Moreover, GBSO is compared against the 2011 version of Standard PSO (SPSO) [17], Global-best guided ABC (GABC) [18] and the Improved Global-best Harmony Search (IGHS) [19] on the CEC05 [20] and the CEC14 [21] benchmarks for increased problem sizes.

The rest of the paper is organized as follows: Section 2 gives details about the BSO algorithm. Different improvements proposed in the literature to improve BSO are covered in Section 3. The proposed

E-mail address: melabd@auk.edu.kw.

http://dx.doi.org/10.1016/j.swevo.2017.05.001
Received 4 August 2016; Received in revised form 17 February 2017; Accepted 5 May 2017

GBSO is fully detailed in Section 4. Section 5 presents the experimental study and the reached results. Finally, the paper is concluded in Section 6.

2. Brain storm optimization

In BSO, a population is defined as a collection of *ideas*. A single *idea* represents a solution to the problem. In each iteration, a population of ideas (solutions) is updated. Initially, ideas are randomly scattered in the search space. In a single iteration, each idea $idea^i$ is updated as follows:

- First, k-means clustering is used to group similar ideas and the best idea in each cluster is saved as the *cluster center*,
- Second, BSO generates a new idea $nidea^i$ by setting it equal to one of the following:
  - A probabilistically selected cluster center,
  - A randomly selected idea from a probabilistically selected cluster,
  - The random combination of two probabilistically selected cluster centers, or
  - The random combination of two randomly selected ideas from two probabilistically selected clusters.

  One of these four operations is randomly selected based on a number of parameters $p_{one-cluster}$, $p_{one-center}$, and $p_{two-centers}$. Moreover, a cluster is probabilistically selected according to its size (i.e. the number of ideas in the cluster),
- Third, the generated $nidea^i$ is perturbed using a step-size parameter ξ and Gaussian distribution,
- Finally, $nidea^i$ replaces the current $idea^i$ if it has a better fitness. Otherwise, it will be discarded.

The BSO algorithm is shown in Fig. 1.

Algorithm 1. The BSO algorithm.

Require $MaxIterations$, n, m, $p_{one-cluster}$,$p_{one-center}$, and $p_{two-centers}$
Randomly initialize n ideas
Evaluate the n ideas
$iter$=1
while $iter \leq MaxIterations$ **do**
  Cluster n ideas into m clusters using k-means
  Rank ideas in each cluster and select cluster centers
  foreach idea i **do**
  if $rand < P_{one-cluster}$ **then**
  Probabilistically select a cluster c_r
  if $rand < P_{one-center}$ **then**
  $nidea^i = center_{c_r}$
  else
  Randomly select an idea j in cluster c_r
  $nidea^i = idea^j_{c_r}$
  end if
  else
  Probabilistically select two clusters c_{r1} and c_{r2}
  Randomly select two ideas c^j_{r1} and c^k_{r2}
  r=$rand$
  if $rand < P_{two-centers}$ **then**
  $nidea^i = r \times center_{c_{r1}} + (1 - r) \times center_{c_{r2}}$
  else
  $nidea^i = r \times idea^j_{c_{r1}} + (1 - r) \times idea^k_{c_{r2}}$
  end if
  end if
  $\xi = rand \times logsig(\frac{0.5 \times Max.Iterations - Current.Iteration}{k})$
  $nidea^i = nidea^i + \xi \times N(0, 1)$
  iffitness($nidea^i$) > fitness($idea^i$) **then**
  $idea^i$=$nidea^i$
  end if
  end for
  end while
  returnbest idea

Note that n is the population size, m is the number of clusters, $N(0, 1)$ represents a Gaussian distribution with mean 0 and a standard deviation of 1. *rand* is a uniformly distributed random number between 0 and 1. Finally, ξ is a dynamically updated step-size and k is for changing the slope of the *logsig* function. For more on BSO, interested readers can refer to [22].

3. Previous BSO improvements

A number of improvements have been proposed in the literature to improve the performance of BSO by addressing some of its disadvantages.

3.1. The clustering process

To overcome the burden of the clustering process, the authors in [9] used a Simple Grouping Strategy (SGM) instead of k-means clustering. In their approach, m seeds are selected randomly at each iteration, and then each one of the n ideas in the current population are assigned to the group of the nearest seed. In addition, perturbing the newly generated idea was done using an idea difference approach. Their method replaced the ξ parameter with a different factor p_r that controls injecting the open minded element, represented by randomly generated problem variables, into the idea creation process. Although their grouping strategy reduced the computational burden of k-means, it still requires a lot of distance calculations in the search space in order to assign different ideas to the different groups. Moreover, their grouping strategy has slightly sacrificed the performance on multi-modal functions. The proposed algorithm provided better results over PSO, DE, and BSO on a small set of classical functions.

Another attempt to overcome the burden of the clustering process was reported in [10]. The work introduced the idea of random grouping to minimize the clustering overhead. This is done by dividing the population into randomly constructed m clusters choosing the fittest idea in each group as its center. Although their grouping strategy aimed at mimicking random discussions between human individuals, it does not provide any ground basis or similarity measures for clustering. In other words, the random grouping strategy could be regarded as the exact opposite of k-means while the SGM previously explained lies somewhere in the middle.

To overcome the challenge of successfully presetting an appropriate number of clusters, the authors in [23] proposed to dynamically set the number of clusters using Affinity Propagation (AP) clustering. AP continuously changes the number of clusters according to their structure information. Although AP clustering still requires some computational effort as similarities need to be calculated between every two data points, the authors did not comment on the computational cost of their algorithm in comparison to using k-means. In addition, no information was given about how the *preference* for AP clustering was set although it has a direct effect on the generated number of clusters. Authors provided experiments showing how their algorithm provides good deployment of a Wireless Sensor Network.

Yet another approach to improve the clustering stage in BSO was recently proposed in [11]. The authors used Agglomerative Hierarchical Clustering (AHC) as it does not require pre-specifying the number of clusters. Moreover, the probability of creating new ideas using a single or multiple (two or three) clusters is adapted according to the quality of solutions generated. However, the authors did not provide details on how these individuals are generated. The developed