Applied Mathematics and Computation 241 (2014) 267–282

Contents lists available at ScienceDirect

Applied Mathematics and Computation

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

A self-adaptive combined strategies algorithm for constrained optimization using differential evolution

Saber M. Elsayed *, Ruhul A. Sarker, Daryl L. Essam

School of Engineering and Information Technology, University of New South Wales at Canberra, Australia

ARTICLE INFO

Keywords:
Differential evolution
Constrained optimization
Evolutionary algorithms
Multi-operator algorithms

ABSTRACT

There are a huge number of differential evolution variants that have been proposed in the literature for solving constrained problems. However, none of them was considered as being a well-accepted approach for solving a broad range of problems with different mathematical properties. Therefore, in this paper, for a better coverage of the problem characteristics, a self-adaptive differential evolution algorithm is introduced. To do that, it uses multiple search operators in conjunction with multiple constraint handling techniques. The need for such an approach is justified by experimental analysis on a well-known set of problems. The results show that the proposed algorithm is superior to other state-of-the-art algorithms.

1. Introduction

There is no doubt that there are a significant number of real-world decision processes that require solutions to optimization problems, which can be defined as finding the best solution among all feasible solutions. Problems that need to optimize an objective function of a number of variables, subject to satisfying certain constraints, are called constrained optimization problems (COPs). Generally speaking, a COP is declared as follows:

$$min\, f(\vec{X})$$

Subject to:

$$\begin{aligned} & g_k(\vec{X}) \leqslant 0, \quad k = 1, 2, \ldots, K \\ & h_q(\vec{X}) \leqslant 0, \quad e = 1, 2, \ldots, Q \\ & \underline{L}_j \leqslant x_j \leqslant \bar{U}_j, \quad j = 1, 2, \ldots D \end{aligned} \tag{1}$$

where $\vec{X} = [x_0, x_1, \ldots, x_D]^T$ is a vector with D-decision variables, $f(\vec{X})$ is the objective function, $g_k(\vec{X})$ is the kth inequality constraints, $h_e(\vec{X})$ is the eth equality constraint, and where each x_j has a lower limit $\underline{L}_j$ and an upper limit $\bar{U}_j$.

Researchers and practitioners use both conventional optimization approaches [1] the computational intelligence (CI) methods [2–4] to solve COPs. One drawback of conventional optimization methods is that as they both require the satisfaction of specific mathematical properties (such as convexity, continuity and differentiability), they may require simplification of the problem and the also making of various assumptions for convenience in mathematical modeling [5]. To add to this, as

* Corresponding author.
E-mail addresses: s.elsayed@adfa.edu.au (S.M. Elsayed), r.sarker@adfa.edu.au (R.A. Sarker), d.essam@adfa.edu.au (D.L. Essam).

http://dx.doi.org/10.1016/j.amc.2014.05.018
0096-3003/

they are not robust to dynamic changes in the environment, they often require a complete restart in order to provide a solution. In contrast, CI algorithms are simple in concept, do not require the satisfaction of specific mathematical properties, are robust to dynamic changes, can handle evaluating solutions in parallel, have the capability for self-organization and have broader practical applications [6].

Evolutionary algorithms (EAs) are well-known CI methods. There are several EAs, such as genetic algorithm (GA) [7,8] and differential evolution (DE) [3,9–11]. However, because of the variations of the mathematical properties of optimization problems, there is no guarantee that any EA will work on all types of problems. This aspect of problem solving difficulty is consistent with the no free lunch (NFL) theorem [12].

To deal with the above mentioned problem, a variety of algorithms have been proposed. One of the successful algorithms is multi-operator EAs [13–15]. Elsayed et al. [16] introduced a general framework for solving COPs, in which each combination of search operators (a mutation strategy with a crossover operator) had its own subpopulation and the subpopulation sizes varied adaptively, as the evolution progresses, depending on the reproductive success of the search operators. In their proposed adaptive approach, they proposed a measure for reproductive success, based on fitness values and constraint violations. The algorithm was then further analyzed and extended in [17], in which a deep analysis was done and a local search procedure was used. Elsayed et al. [18] proposed two variants of DE algorithms, each of them used a combination of several mutation and crossover operators. The algorithms were tested on a set of 36 test problems, and showed better performance than other state-of-the-art algorithms.

In the literature, Qin et al. [19] proposed the Self-adaptive Differential Evolution algorithm (SaDE). They have used two mutation strategies, where each individual is assigned to one of them based on a given probability. After evaluation of all newly generated trial vectors, the numbers of trial vectors successfully entering the next generation were recorded as ns_1 and ns_2. Those two numbers were accumulated within a specified number of generations, called the "learning period". Then, the probability of assigning each individual was updated. However, in their algorithm, a strategy might be totally excluded from the list if its p became equal to 0. Yong et al. [20] developed a composite DE algorithm (CoDE), wherein a new trial vector was created by randomly combining three trial DE strategies with three control parameter settings at each generation. Thus, three trial vectors were generated for each target vector, and the best among them made its way to the population of the next generation if it was better than its target vector. The algorithm demonstrated competitive performance on a set of unconstrained test problems. Yang et al. [21] introduced a self-adaptive clustering-based DE with composite trial vector generation strategies (SaCoCDE), in which the population was divided into different subsets by a clustering algorithm. The algorithm was tested on a set of unconstrained problems and showed highly competitive performance when compared with the state-of-the-art DE algorithms.

Zhang et al. [22] introduced an adaptive differential evolution algorithm with optional external memory (JADE). In it, at each generation, Cr_z of each individual, x_z, was independently generated according to a normal distribution of mean μCr and standard deviation of 0.1. μCr was initialized at a value of 0.5 and was updated. Similarly, F_z of each individual x_z was independently generated according to a Cauchy distribution with location parameter μF and scale parameter 0. The location parameter μF was initialized to 0.5 and was subsequently updated at the end of each generation.

Zamuda and Brest [23] proposed an algorithm that incorporated two multiple mutation strategies into jDE, and introduced a population reduction methodology in [24]. The algorithm was tested on 22 real-world applications. The algorithm showed better performance over two other algorithms.

Mallipeddi et al. [25] proposed an ensemble of mutation strategies and control parameters with DE (EPSDE) for solving unconstrained optimization problems. In EPSDE, a pool of distinct mutation strategies, along with a pool of values for each control parameter, coexists throughout the evolution process and competes to produce offspring. Mallipeddi et al. [26] also proposed an algorithm that uses an ensemble of different constraint handling techniques for solving constrained problems. The proposed algorithm has shown good performance in comparison to other algorithms.

Tasgetiren et al. [27] proposed an ensemble DE, which worked in such a way that each individual was assigned to one of two distinct mutation strategies or a variable parameter search (VPS). VPS was used to enhance the local exploitation capability. However, no adaptive strategy was used in that algorithm. Tasgetiren et al. [28] proposed a discrete DE algorithm with a mix of parameter values and crossover operators to solve a traveling salesman problem, in which parallel populations were considered. Each parameter set and crossover operator was assigned to one of the parallel populations. Furthermore, each parallel parent population competed with the same population's offspring as well as the offspring populations generated by all other parallel populations. The algorithm has shown improved results in comparison to other state-of-the-art-algorithms. However, the proposed algorithm was computationally at least twice more expensive than the other algorithms considered in the paper.

Based on the literature, there was no single DE that used a mix of all mutation and crossover operators as well as several constraint handling techniques. Therefore, in this research, a differential evolution algorithm with self-adaptive multi-combination strategies is proposed. We named the proposed algorithm as SAS-DE. SAS-DE combines the strengths of four mutations, two crossover operators, and two constraint handling techniques. In SAS-DE, each individual is first assigned to a random combination of search operators and constraint handling techniques. The algorithm is analyzed by solving a set of benchmark problems [29]. In testing, the algorithm showed consistently better performance in comparison to other state-of-the-art algorithms.

We wish to mention here that our work is different than [25], in that: (1) we have used an improvement scheme to decide, adaptively, the preferable combinations of strategies, (2) to get the benefit of the whole population's information, we have not divided the population into subpopulations, (3) we have used a self-adaptive formula to generate the DE