Expert Systems with Applications 40 (2013) 3370–3377

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

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

Investigating Multi-View Differential Evolution for solving constrained engineering design problems

Vinícius V. de Melo [a,*], Grazieli L.C. Carosio [b]

[a] Institute of Science and Technology, Federal University of São Paulo, São José dos Campos, SP, Brazil
[b] Instituto de Aeronáutica e Espaço, Depto de Ciência e Tecnologia Aeroespacial, São José dos Campos, SP, Brazil

ARTICLE INFO

Keywords:
Evolutionary computations
Global optimization
Constrained optimization
Metaheuristics
Differential Evolution

ABSTRACT

Several constrained and unconstrained optimization problems have been adequately solved over the years thanks to advances in the metaheuristics area. In the last decades, different metaheuristics have been proposed employing new ideas, and hybrid algorithms that improve the original metaheuristics have been developed. One of the most successfully employed metaheuristics is the Differential Evolution. In this paper it is proposed a Multi-View Differential Evolution algorithm (MVDE) in which several mutation strategies are applied to the current population to generate different views at each iteration. The views are then merged according to the winner-takes-all paradigm, resulting in automatic exploration/exploitation balance. MVDE was tested to solve a set of well-known constrained engineering design problems and the obtained results were compared to those from many state-of-the-art metaheuristics. Results show that MVDE was very competitive in the considered problems, largely outperforming several of the compared algorithms.

1. Introduction

Over the years, several metaheuristics have been proposed to solve constrained problems by finding global optimum solutions. Some are hybrid approaches (He & Wang, 2007b; Liu, Cai, & Wang, 2010; Michalewicz & Schoenauer, 1996; Pedamallu & Ozdamar, 2008), many are classical algorithms with new operators or improvements (Coelho, 2010; Coello & Mezura-Montes, 2002; He & Wang, 2007a; Kimbrough, Koehler, Lu, & Wood, 2008; Pant, Thangaraj, & Abraha, 2009), others are self-adaptive version of classical algorithms (Coello Coello, 2000; Mezura-Montes, Coello, & Velázquez-Reyes, 2006; Michalewicz & Schoenauer, 1996; Noman & Iba, 2008).

An improved EA, named Differential Evolution (DE) (Storn & Price, 1997), was presented as an effective, robust, and simple global optimization algorithm which has only a few control parameters. Many works have shown that DE outperforms many other optimization methods, in terms of convergence speed and robustness, in solving hard benchmark functions and real-world problems (Chakraborty, 2008). A recent and very complete review can be seen in Das and Suganthan (2011).

DE has only three parameters: population size, the amplification factor (F) and the crossover probability (CR). Choosing an adequate configuration depends on the problem and on the mutation and crossover operators. Based on these issues, several variants have been proposed to improve DE in a self-adaptive way, employing several strategies at the same time and dynamic adjustment of parameter control over the generations to perform better exploration and exploitation of the search-space (Qin, Huang, & Suganthan, 2009; Wang, Cai, & Zhang, 2011; Zhang & Sanderson, 2009).

In general those approaches combine several strategies with several control parameter settings, according to percentage of success in replacing the parent solution, to generate a single population. After the population is evaluated, the percentages of selection from the possible combinations are updated. Thus, in the following iterations some combinations get a higher possibility of generating children than others.

In this paper a similar approach was taken, but without calculating percentages and using a different combination of the trial solutions. Also, for the current algorithm there is no self-adaptation of the control parameters. The proposed approach, called Multi-View Differential Evolution (MVDE) is a simple modification of the DE algorithm and thus requires a low effort to be implemented.

To evaluate MVDE's performance, the algorithm was employed to solve five well-known constrained engineering design problems, and the results were compared to those from several state-of-the-art algorithms.

This paper is organized as follows. In Section 2 the Differential Evolution algorithm is briefly introduced. The new algorithm, MVDE, is elaborated in Section 3. Section 4 introduces constrained

* Corresponding author. Tel.: +55 12 3309 9500.
E-mail addresses: vinicius.melo@unifesp.br (V.V. de Melo), grazieliglcc@iae.cta.br (G.L.C. Carosio).

0957-4174/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2012.12.045

optimization and presents the numerical examples (engineering problems), details of the experiments, the results obtained, and the discussion. Finally, in Section 5 some conclusions are drawn about the results.

2. Differential Evolution

Differential Evolution was introduced by Storn and Price (1995). It is a real-valued populational metaheuristic that works like evolutionary algorithms, successfully used to solve several benchmarks and real-world problems (Chakraborty, 2008; Melo & Delbem, 2009; Neri, Iacca, & Mininno, 2011; Pan, Wang, Gao, & Li, 2011; Wang, Li, & Weise, 2010).

The basic functioning is as follows. A population P with N vectors of D dimensions is randomly initialized (using a uniform distribution) inside the problem's bounds and evaluated using the objective/fitness function for the problem. Then, until a stop condition is satisfied, the algorithm performs an iterative evolutionary process of mutation, crossover and selection operations.

For each vector x_i from P, the mutation operator uses the weighted difference of parent solutions to generate trial vectors v_i. In this work the following mutation strategies were selected:

1. **rand/1**
$$v_i = x_{r1} + F \times (x_{r2} - x_{r3}) \tag{1}$$
2. **best/1**
$$v_i = x_{best} + F \times (x_{r2} - x_{r3}) \tag{2}$$
3. **current-to-best/1**
$$v_i = x_i + F \times (x_{best} - x_i) + F \times (x_{r1} - x_{r2}) \tag{3}$$
4. **best/2**
$$v_i = x_{best} + F \times (x_{r1} - x_{r2}) + F \times (x_{r3} - x_{r4}) \tag{4}$$
5. **rand/2**
$$v_i = x_{r1} + F \times (x_{r2} - x_{r3}) + F \times (x_{r4} - x_{r5}) \tag{5}$$

where $x_{r1}, x_{r2}, x_{r3}, x_{r4}$, and x_{r5} are five distinct and randomly chosen vectors from P, x_{best} is the best solution from P, and $F \in [0,2]$ is the *mutation or amplification factor*. Classically, the binomial crossover operator is applied on v_i to generate the final offspring vector u_i according to

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if } U \sim (0,1) \leqslant CR \text{ or} \\ & j = trunc(U \sim (1,D)), \\ x_{i,j}, & \text{otherwise} \end{cases} \tag{6}$$

where $j = 1, \ldots, D$; $U(a,b)$ is a random floating-point number from a uniform distribution between a and b generated for each j, and $CR \in [0,1]$ is the crossover probability.

Finally, the selection step selects the best evaluated vector between x_i and u_i. The offspring replaces the parent if its fitness value is better. Otherwise, the parent is maintained in the population.

3. Multi-View Differential Evolution

In this work, the Multi-View learning (Chen & Yao, 2008; Crammer, Kearns, & Wortman, 2008) is proposed as a metaheuristic enhancement and is directly applied to improve DE. In Machine Learning Mitchell (1997), Multi-View learning is a paradigm in which a learning algorithm uses the agreement among multiple learners to decide about a prediction. Multiple hypotheses are trained from the same dataset – where the instances are known (labeled) - to generate predictions on one or more unlabeled examples. In a classification process each hypothesis (called *view*) may be a different algorithm (neural networks, SVM, decision-trees, etc.) or the same algorithm with different settings. Each *view* presents a response/prediction for the unlabeled example to be classified. A voting procedure, for instance, is then employed to decide the winner prediction among the *views*.

Based on that idea, the algorithm proposed in this work is named Multi-View Differential Evolution (MVDE). Instead of using several populations, sub-populations, or co-evolution, the idea proposed in this work consists of employing different strategies to generate new trial solutions from the *same* population, thus providing different *views* for the same problem. Different *views* lead to exploration of different regions. Some strategies generate solutions toward a local optimum whereas other strategies are better in escaping from a local optimum areas. No self-adaptation is employed.

However, differently from the approaches employed in other similar algorithms (Qin et al., 2009; Wang et al., 2011; Zhang & Sanderson, 2009) that change the number new of trial vectors for each strategy to maintain the population size, in MVDE N trial vectors are always generated for each view which are then merged to be the selected population of trial vectors. The proposed algorithm is presented in Algorithm 1.

Algorithm 1. Algorithm of MVDE.

```
Generate a large population
Evaluate the population
Select best N solutions to be the actual initial
  population
Initialize views(1 to v)
Do
/* Generate the views: */
  For each view
    Apply, in the current population, the mutation
  strategy corresponding to the current
    view and save it in the view's population
    Evaluate the vector generated
  End for
/* Elitism: */
  Save current best solution and the best vector of
  all views in the merged
  mutated population
/* Merge the views: */
  For index = 1 to N in the merged mutated population
    For each view
      Get the vector from position index
    End for
    Apply tournament-selection in the taken
  vectors
    Save the winner in the merged mutated population
  End for
  Remove the two worst vectors from the merged
  mutated population
/* Usual crossover */
  Apply crossover to the merged mutated population
Until the termination condition is reached
```

Three modifications in the classical DE are proposed. First of all, a large sample of points in the search-space is created. The fitness is calculated and the N best solutions are selected to be MVDE's population, as proposed in Melo and Delbem (2008). This allows for a better initial coverage and possibly unbiased starting sampling of the search-space.