Contents lists available at ScienceDirect

Applied Soft Computing

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

Training and testing a self-adaptive multi-operator evolutionary algorithm for constrained optimization

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

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

A R T I C L E I N F O

Article history:
Received 5 September 2012
Accepted 12 October 2014
Available online 24 October 2014

Keywords:
Constrained optimization
Genetic algorithm
Cross validation

A B S T R A C T

Over the last two decades, many different evolutionary algorithms (EAs) have been introduced for solving constrained optimization problems (COPs). Due to the variability of the characteristics in different COPs, no single algorithm performs consistently over a range of practical problems. To design and refine an algorithm, numerous trial-and-error runs are often performed in order to choose a suitable search operator and the parameters. However, even by trial-and-error, one may not find an appropriate search operator and parameters. In this paper, we have applied the concept of training and testing with a self-adaptive multi-operator based evolutionary algorithm to find suitable parameters. The training and testing sets are decided based on the mathematical properties of 60 problems from two well-known specialized benchmark test sets. The experimental results provide interesting insights and a new way of choosing parameters.

1. Introduction

Constrained optimization is a challenging research area in the computer science, and optimization domains. In order to solve the complex decision problems, many problems are defined as constrained optimization problems (COPs). COPs can be divided into many different categories based on their characteristics and mathematical properties.

Evolutionary algorithms (EAs) have a long history of successfully solving optimization problems regardless of whether or not they have nice mathematical properties. The EAs family contains a wide range of algorithms that have been used to solve optimization problems, such as the genetic algorithm (GA) [1,2], differential evolution (DE) [3], evolution Strategies (ES) [4], evolutionary programming (EP) [5]. A comparative study among some of these EAs is found in [6]. In this research, we consider DE for solving optimization problems, because DE usually converges quickly, incorporates a relatively simple and self-adapting mutation, and the same settings can be used for many different problems [3]. EA has the ability to deal with both continuous and discrete variables, and can deal with extremely complex fitness landscapes, including under noisy and dynamic environments.

Due to the variability of problem characteristics, and their underlying mathematical properties, an algorithm that demonstrated to work well for one problem, or a class of problems, does not guarantee that it will work for another problem or a range of problems. This behavior can be argued as consistent with the no free lunch (NFL) theorem [7]. For this reason, new algorithms are continuously reported in the literature for solving new problems with different characteristics. In developing any new evolutionary algorithm, it is required to decide on two main aspects: (i) designing an appropriate algorithm and (ii) choosing the required algorithmic parameters. In reality, the algorithm design is not a single step process, but rather an iterative process. After the initial design, the algorithm is redesigned and refined until it meets some criterion. In most cases, the criterion is set as the quality of the solutions for a given set of test problems. Once the criterion is met, only the final form of the algorithm is reported. The parameters to be used in the algorithm are usually determined through learning from the test problems. Traditionally, numerous trial-and-error runs are usually performed to choose suitable parameters. However, even by trial-and-error, one may not find appropriate parameters. Such a parameter selection process is sometimes called parametric analysis. We can now argue that an algorithm with the chosen parameters may demonstrate adequate solution quality on the test problems that may not perform well with unknown problems. However, in evolutionary optimization domain, the usual practice is to compare the performance of any new algorithm with the existing algorithm(s) through solving a set of benchmark problems.

* Corresponding author. Tel.: +61 2 626 88817.
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.asoc.2014.10.011
1568-4946/© 2014 Elsevier B.V. All rights reserved.

From the above discussions, we can state that there is no single algorithm or an algorithm with a known search operator and given parameters, that will consistently perform best for all classes of optimization problems. This motivates us to consider multiple search operators for a better coverage of problems and a train-and-test based approach for appropriate parameter selection.

Murata and Ishibuchi [8] examined the performance of different variants of GA with two types of genetic operators (each variant with one crossover and one mutation) in solving flow shop scheduling problems. As they have shown, an independently evaluated good operator may not perform well when it is used in conjunction with another operator (here a given crossover with a given mutation). They also revealed that the combined effect of two operators, which determines the effectiveness of the algorithm, could be either positive or negative. This means that the choice of operators is important in designing a high performing GA, further, this choice is often made by trial-and-error. In this research, our focus is on multi-operator based evolutionary algorithms, where multiple operators, of differing types, will be considered together within the framework of one algorithm. This type of algorithm can be seen not only as a better alternative over trial-and-error based designs, but also as a means to generate better coverage of problems. Multi-operator based evolutionary algorithms are not new in the literature. However, their actual ability for solving constrained optimization problems has not been fully explored. Also, the choice of operators and their appropriate mix, and strategies for their effective use, has not been well studied. A brief review of multi-operator based EAs is provided below.

In the case of multiple operators based EAs, it is a popular practice to use adaptive approach. For example, multi-operator based EAs are not new in the literature. In multiple operators based EAs, it is a popular practice to use an adaptive approach. Spears [9] applied an adaptive strategy using two different crossovers for solving N-Peak problems. Eiben [10] developed an adaptive GA framework with multiple crossover operators for solving unconstrained problems. In their algorithm, the population was divided into a number of sub-populations, each of which used a particular crossover. Based on the success of the crossovers, the sub-population sizes were varied. However, their adaptive GAs did not outperform the standard GA using only the best crossover. Hyun-Sook and Byung-Ro [11] investigated whether a combination of crossover operators could outperform the usage of only the best crossover operator by solving the TSP and the graph bisection problem. They used an adaptive strategy to assign the probability of using different crossovers.

In DE, Mallipeddi et al. [12] proposed an ensemble of mutation strategies and control parameters with DE (EPSDE), in which a pool of distinct mutation strategies, along with a pool of values for each control parameter, coexisted throughout the evolutionary process and competed to produce offspring. This algorithm was used to solve a set of unconstrained problems. Mallipeddi and Suganthan [13] proposed using a mix of four constraint handling techniques (CHTs) based on a DE algorithm (ECHT-DE) to solve COPs in which different populations were initialized and such that one of each CHT was assigned to each population. Furthermore, mixes of mutation strategies and amplification factor (F) and crossover rate (Cr) values were also used, along with the two mutation strategies "DE/rand/2/bin" and "DE/current-to-rand/1/bin". The pools of Cr and F values were in the ranges of 0.1–0.9 and 0.4–0.9, respectively, and in steps of 0.1. This algorithm came second in the CEC2010 competition for COPs. However, it was expensive in terms of computational time. Mallipeddi and Suganthan [14] extended their previous work to solve a set of real-world applications [15]. However, their obtained results were not very good in comparison with those from other algorithms. Tasgetiren et al. [16] proposed a discrete DE algorithm with a mix of parameter values and crossover operators, in which parallel populations were initialized, to solve the TSP. 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 of the other parallel populations. Although this algorithm showed better results than other state-of-the-art-algorithms, it was computationally at least twice as expensive as those against which it was compared in the paper. Yong et al. have recently proposed a composite DE algorithm (CoDE) [17], in which the algorithm randomly combines several trial vector generation strategies with a number of control parameter settings at each generation to create new trial vectors. CoDE has been tested on a set of unconstrained problems and showed competitive performance in comparison to other state-of-the-art algorithms. Mallipeddi et al. [18] proposed an EP algorithm that used different mutation strategies, in which each mutation operator has its associated population and every population benefits from every function call. The algorithm has shown superior performance in comparison to other EP algorithms. Elsayed et al. [19] proposed a mix of four different DE mutation strategies within a single algorithm framework to solve COPs, and that algorithm showed good performance by solving a set of small scale theoretical benchmark constrained problems. The algorithm was further extended in [20,21]. Elsayed et al. [22] proposed two novel DE variants, in which each variant utilized the strengths of multiple mutation and crossover operators for solving 60 constrained problems. The algorithm showed competitive, if not better, performance in comparison to the state-of-the-art algorithms.

We consider a self-adaptive multi-operator genetic algorithm (SAMO-GA) for the constrained optimization problem, which is much more complex than its unconstrained counterpart. In SAMO-GA, each combination of search operators has its own subpopulation. Further, the subpopulation sizes vary adaptively, as the evolution progresses, depending on the reproductive success of the search operators. However, the sum of the size of all of the subpopulations is fixed during the entire evolutionary process. To do this, three equations have been introduced for determining the reproductive success based on the fitness values and the constraint violations. Lastly, to deal effectively an operator may perform very well at earlier stages of the evolution process and do badly at later stages or vice versa, a lower bound on the size of each subpopulation has been set.

Therefore, the aim of this research is to measure the performance of SAMO-GA on a diverse set of constrained problems, in which we extend the idea of training and testing with EAs for solving COPs, in which we use a set of test problems that contain different properties for their objective functions and constraints [23,24]. Cross-validation [25] is then used to estimate the generalization ability of SAMO-GA. This is done, in such way that the problems are divided into three groups, with the consideration that each group contains the most diverse types of problems, as possible. Each two groups are then used for training, while the third one is used for testing. Finally, the mean square error measure [26] is used to quantify the difference between the obtained results and the best known results.

We could not find any research dealing with the training and testing concept with evolutionary algorithms for the purpose of solving constrained optimization. One of the challenging issues for this concept, is how can we design the training and testing problems? As it is known in machine learning domain, the idea of separating the data into training and testing sets comes with a number of assumptions, such as both training and testing datasets come from the same distribution. In this paper, we have divided the test problems, from two well-known problem sets, into three groups based on their mathematical properties.

This paper is organized as follows. After the introduction, Section 2 describes the design of SAMO-GA, and the constraint