Neurocomputing 275 (2018) 155–166

Contents lists available at ScienceDirect

Neurocomputing

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

An improved class of real-coded Genetic Algorithms for numerical optimization☆,☆☆

Mostafa Z. Ali[a,*], Noor H. Awad[b], Ponnuthurai N. Suganthan[b,*], Ali M. Shatnawi[a], Robert G. Reynolds[c]

[a] Jordan University of Science & Technology, 22110, Jordan
[b] School of Electrical & Electronic Engineering, Nanyang Technological University, Singapore
[c] Wayne State University, Detroit, MI, USA

ARTICLE INFO

Article history:
Received 9 October 2016
Revised 26 December 2016
Accepted 22 May 2017
Available online 26 May 2017

Communicated by Prof. Zidong Wang

Keywords:
Genetic Algorithms
Optimization problem
Differential Evolution
Multi-parent crossover

ABSTRACT

Over the last few decades, many improved Evolutionary Algorithms (EAs) have been proposed to tackle different types of optimization problems. Genetic Algorithm (GA) among other canonical algorithms have not shown consistent performance over a range of different optimization problems with complex characteristics. In this paper, an improved class of real-coded Genetic Algorithm is introduced to solve complex optimization problems. The first algorithm, Genetic Algorithm embedded with a new Differential Evolution crossover, GA–DEx, proposes a new variant of Differential Evolution mutation which is used as a new multi-parent crossover in Genetic Algorithms. The main purpose of this algorithm is to enhance the search ability of the GA algorithm by combining a new Differential Evolution crossover with a GA algorithm to avoid premature convergence and stagnation scenarios by exploring more solutions in the problem search space. The second amalgam algorithm, GA–DEx$_{SPS}$, uses an effective and efficient successful parent selection strategy to provide a successful alternative for the selection of parents during the Differential Evolution crossover process. This strategy improves the performance of first introduced algorithm by selecting more promising parents to guide the evolutionary search. The third algorithm, GA–*a*DEx$_{SPS}$, introduces an aging mechanism and a success-history-based adaptive Genetic Algorithm. This algorithm adapts the *alpha* parameter used by Differential Evolution crossover in a history-based adaptive manner. This adaptation helps the search to discover more promising regions and to prevent stagnation and premature convergence scenarios. To verify the performance of our class, a challenging test suite of 30 benchmark functions from the IEEE CEC2014 real parameter single objective competition is used. The results affirm the effectiveness and robustness of the proposed algorithms compared to other state-of-the-art well-known crossovers and recent Genetic Algorithms variants.

1. Introduction

Evolutionary algorithms (EAs) based upon Darwinian principle are familiar for their ability to handle complex and non-linear optimization and real-world problems. In the presence of different problem characteristics- such as high dimensionality, non-linearity, non-seperability, ill-conditioning, and multimodality-finding the optimal solution is a challenging task. Over the last few decades, many evolutionary algorithms (EAs) have demonstrated enormous success in solving these complex optimization problems. The family of EAs consists of many population-based algorithms such as Genetic Algorithms (GA) [19], Evolutionary Programming (EP), Differential Evolution (DE) [37] and Cultural Algorithms (CA) [36].

The GA was introduced by Holland [19], and is considered a population-based algorithm which consists of three important operators: crossover, mutation and selection. The evolution process starts by initializing a population space of random individuals (chromosomes). In each generation, the fitness of individuals is evaluated based on an objective function of the optimization problem. The fittest individuals are selected to be parents in the next generation. Then each individual is modified based on mutation and crossover operations to form new individuals for the next generation. Finally, the algorithm terminates when either the optimal solution has been found or the maximum number of

☆ This manuscript is the authors' original work and has not been published nor has it been submitted simultaneously elsewhere.
☆☆ All authors have checked the manuscript and have agreed to the submission.

* Corresponding authors.
E-mail addresses: mzali.pn@gmail.com (M.Z. Ali), noor0029@ntu.edu.sg (N.H. Awad), epnsugan@ntu.edu.sg (P.N. Suganthan), ali@just.edu.jo (A.M. Shatnawi), Robert.reynolds@wayne.edu (R.G. Reynolds).

http://dx.doi.org/10.1016/j.neucom.2017.05.054
0925-2312/

generations has been reached [29]. The GA proved to be capable of solving many real-life applications and single optimization problems [7,9,43,53].

The early development of the GA used three different crossover types for binary coded GAs: uniform crossover, two-point, and multi-point crossovers [17]. On the other hand, the two-parent crossover which was widely used in practice [17], and multi-parent crossovers are introduced for real-coded GAs [5,12,30,39,44] . In the two-parent crossover, two parents (p_1, p_2) are used to generate a new offspring (O) that lies between the selected parents for each gene position where p and O are D-dimensional vectors such that: $p = \{p^0, \ldots, p^{D-1}\}$, $o = p = \{o^0, \ldots, o^{D-1}\}$ and D is the dimension of the problem being solved. It is known that three existing arithmetic crossovers proposed in the literature which are: simple arithmetic crossover, single arithmetic crossover and whole arithmetic crossover, and are considered as examples of two parent crossovers [11]. The simple arithmetic crossover chooses a recombination point k at first, then the first k values are taken from a randomly chosen parent and copied into the child. The rest of values are the result of the arithmetic average of the two selected parents according to the chosen random number α. For the single arithmetic crossover, the values of the k positions take the arithmetic average of the two selected parents while the others are copied from the parents. The whole arithmetic crossover is probably considered the most commonly used operator. It works by taking the weighted sum of the two selected parents with the same α for each gene. There are many other types of two-parent crossovers such as: BGA, Average, One point, Heuristic, Discrete, BLX-alpha and BLX-alpha-beta [11].

The multi-parent crossover uses more than two parents to generate a new offspring and hence they have been called multi-parent crossovers. Examples are unimodal distribution crossover (UNDX) [30], simplex crossover (SPX) [44], parent centric crossover (PCX) [5] and triangular crossover (TC) [12]. UNDX uses the center of mass for N multiple parents to generate offspring solutions by assigning small different probability away from the center of mass. Despite the fact that UNDX has shown superb performance when solving highly epistasis problems, it fails to evolve good offspring solutions in two cases. The first one is when the population size is relatively small compared to the problem search space. The other case is when the optimal solution is near the boundaries of the search space [30]. SPX is used in real-coded GAs as a multi-parent recombination operator that uses the property of Simplex in the search space. This crossover type uses a region marked by multiple selected parents to define a restricted search space and a uniform probability distribution to generate a new offspring. The types of problems that SPX is capable of solving are the multi-modal and/or epistasis problems using a suitable number of parents. Tsutsui suggests medium number of parents to solve these problems such as three parents for low dimension problems and four parents for high dimension problems [44]. The SPX fails to solve tightly linked sub-functions [44].

PCX is a self-adaptive multi-parent crossover which uses a large probability rather than the center of selected parents to generate a new solution near each parent. One parent is selected for each generated offspring and a difference vector is calculated between this parent and the N chosen parents. Besides, the PCX have difficulties to solve multimodal problems. One of its major issues is the time complexity in comparison to the other crossover operators. TC crossover uses three parents to generate three different offspring solutions using a linear combination of the chosen parents where two of them must be feasible while the other one is infeasible. This type of crossover works well on problems that have single bounded feasible region in the continuous search space and the optimal solution lies on the boundary of this feasible region [12].

In fact, the crossover operation of GA has a major role in the evolution of the search and the mutation was generally treated as subordinate to crossover [19]. From this point of view, many researchers adhere that crossover is the most important operator in GA [31]. Mutation works as a variation or diversity operator that changes some parts of the chromosome based on a predefined probability. The mutated offspring may move to the nearest optimal solution if GA has some kind of adaptation to learn suitable values. Hence, the fitness of the mutated offspring may be better or worse than its parents. Several types of mutations have been proposed in the literature. For real coded GAs, uniform and non-uniform mutations are introduced [29].

In this paper, a new class of improved GAs is introduced to overcome the shortcomings of real-coded GAs when solving complex optimization problems with diverse characteristics. The paper proposes three different approaches. The main objective of the first approach is to improve the most important operator in GA which is the crossover operator. In this approach, namely GA–DEx, a new multi-parent crossover is introduced by merging the strengths and capabilities of Differential Evolution in the GA crossover operation. The algorithm introduces a new variant of DE/rand/1 as a multi-parent crossover in GA. The second approach, GA–DEx$_{SPS}$, introduces a successful parent selection for the proposed crossover in our first approach to improve the search ability by selecting new promising parents to guide the evolution of the search. The third algorithm, GA–aDEx$_{SPS}$, introduces an aging mechanism and a success-history-based adaptive GA where the parameters used in the proposed crossover, DEx, is adjusted adaptively using Cauchy distribution. This history-based process increases the effectiveness of the search during the optimization process by adaptively selecting the suitable parameter settings based on the success of previous generations. To better assess the performance of this proposed crossover, an extensive comparison is performed using well-known two-parent and multi-parent crossover based GA algorithms, among others. The results show the effectiveness and robustness of the proposed crossover among all other existing crossovers and introduced techniques found in the literature.

Many researchers are devoted to develop new GAs to enhance the performance of real-coded GA by improving the search ability to solve premature convergence and stagnation scenarios. Those researchers use three different categories to develop their techniques. The following sections present these three categories and provide some examples. The rest of the paper is organized as follows. Section 2 introduces a review of improved GAs in literature. Section 3 presents the three proposed algorithms in details. Simulation results are presented in Section 4 for the comparison of the three proposed algorithms with other well-known crossovers, recent GAs and powerful DE algorithms. Finally, Section 5 draws conclusions of this work.

2. Review of previous literature on improving the GA algorithm

In the last few decades, many improved GA versions have been proposed in the literature to enhance the GA algorithm performance. Such works can be categorized into three main groups. The first group develops a memetic version of GA as hybrid approaches which merge GA with local search methods or other evolutionary algorithms to fine tune the search [23,31,35]. The second group develops a self-adaptive GA which is another promising and significant approach of enhancing GAs [1,27,30–32]. In such algorithms, the crossover and mutation parameters are adapted instead of using fixed values in each generation. The third group uses an ensemble of different mutation and crossover techniques or a hybridization of the first two groups [22,38,49,58]. In this section, some recent GA-based techniques from these categories that were developed for numerical optimization, are introduced.