Journal of Computational Design and Engineering 4 (2017) 158–167

Contents lists available at ScienceDirect

Journal of Computational Design and Engineering

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

Fluid Genetic Algorithm (FGA) ☆

Ruholla Jafari-Marandi *, Brian K. Smith

Department of Industrial and Systems Engineering, Mississippi State University, 260 McCain Engineering Building, Mississippi State, MS 39762, United States

A R T I C L E I N F O

Article history:
Received 21 April 2016
Accepted 8 March 2017
Available online 9 March 2017

Keywords:
Genetic Algorithm (GA)
Metaheuristics
Non-exact optimization

A B S T R A C T

Genetic Algorithm (GA) has been one of the most popular methods for many challenging optimization problems when exact approaches are too computationally expensive. A review of the literature shows extensive research attempting to adapt and develop the standard GA. Nevertheless, the essence of GA which consists of concepts such as chromosomes, individuals, crossover, mutation, and others rarely has been the focus of recent researchers. In this paper method, Fluid Genetic Algorithm (FGA), some of these concepts are changed, removed, and furthermore, new concepts are introduced. The performance of GA and FGA are compared through seven benchmark functions. FGA not only shows a better success rate and better convergence control, but it can be applied to a wider range of problems including multi-objective and multi-level problems. Also, the application of FGA for a real engineering problem, Quadric Assignment Problem (AQP), is shown and experienced.

1. Introduction

Genetic Algorithm (GA) is a very powerful meta-heuristic and evolutionary algorithm which has been used (Azadeh et al., 2013; Oujebbour, Habbal, Ellaia, & Zhao, 2014; Qu, Liu, Duan, & Yang, 2016), developed (Tavakkoli-Moghaddam & Jafari-Marandi, 2013), adapted (Jafari-Marandi, Hu, & Chowdhury, 2015; Jafari-Marandi, Hu, & Omitaomu, 2016; Keramatia et al., 2014), and hybridized with other evolutionary algorithms (Rabbani, Baghersad, & Jafari, 2013) for many different problems in different disciplines. The algorithm was proposed by John Holland and his colleagues in the 1960s. Of course, there are different variations to the algorithm. Srinivas and Patnaik (1994) carried out the latest literature survey on the algorithm revealing different efforts in trying to improve the algorithm and mitigate its drawbacks. Not long after, the research on this algorithm, similar to any other evolutionary algorithms (Van Veldhuizen & Lamont, 2000), took a turn toward multi-objective optimization (Deb, 1999). Since then there have been great developments in the path of adapting GA for multi objective and multi-level problems. In fact, NSGA-II (Deb, Pratap, Agarwal, & Meyarivan, 2002), a very powerful tool to tackle multi-objective problems, still remains among these great developments.

1.1. Literature review

Metaheuristics are solution procedures that use higher level strategies to enhance existing local improvement strategies with the hope of escaping from local optima and reaching a robust search of a solution space (Glover & Kochenberger, 2006). Their inception goes back to the early 1980s (Osman & Kelly, 1996) and ever since they have been among the most applied and under development areas of engineering and science (Talbi, 2009). Genetic Algorithm and genetic programming remain among highly applied solution methods. GA's well-known advantages are robustness and usability (Salomon, 1996). Research on Genetic Algorithms has taken three different directions. First, because of its great adaptability, researchers have adapted GA to solve different problems in different disciplines. Second, GA is a very popular candidate for being hybridized with other techniques for more improvements. Lastly, due to the importance of using a fine tuned GA to solve problems, behavioral parameters of GA are the focus of researchers' scrutiny.

The contributions of Genetic Algorithm in many areas of science and engineering is undisputable. Solving complex mathematical modeling and optimization problems are among these popular uses. Kakandikar and Nandedkar (2016) adopt GA to solve their complex thinning optimization problem and through experience they show that their genetic coding was successful in dealing with the challenge. Furthermore, Zhang and Zhao (2015) apply a classic GA to approach a special point-to-point transfer time traveling schedule problem. In Azadeh et al. (2013) GA is used to nearly optimally predict missing values in a randomized block design table. GA

☆ Peer review under responsibility of Society for Computational Design and Engineering.

* Corresponding author.
E-mail address: Rj746@msstate.edu (R. Jafari-Marandi).

http://dx.doi.org/10.1016/j.jcde.2017.03.001
2288-4300/

even is employed in data analytic tasks such as clustering (Maulik & Bandyopadhyay, 2000), classification (Uysal & Gunal, 2014) and association rules (Minaei-Bidgoli, Barmaki, & Nasiri, 2013). Additionally, GA's adaptability makes it very popular for solving multi-objective or multi-level (distributed decision making) problems. NSGA-II (Deb et al., 2002), doubtless is among the strongest techniques to approach a multi-objective optimization problem. In fact, Konak, Coit, and Smith (2006) put forward a very popular tutorial on how to use GA for multi-objective optimizations. For example, multi-objective GA (NSGA-II) is used to solve the challenging time and space assembly line under uncertain demand (Chica, Bautista, Cordón, & Damas, 2016). GA is routinely used for multi-level problems. Jafari-Marandi et al. (2016) adapt GA for solving distributed decision making of building clusters. Moreover, Long (2016), in the area of supply chain management, tweaks GA to approach a multi-level collaborative decision making problem.

In the meta-heuristic research community there is a strong surge toward introducing new techniques and methods with the hope of defeating the existing ones. For instance, very recently, Lion Optimization Algorithm (LOA) Yazdani & Jolai, 2016, based on the group dynamic behavior of lions such as prey capturing, mating, and territorial marking, was developed. The efforts of scholars in improving the techniques are in many cases attracted toward hybridizing existing techniques, and GA is among the very popular methods that researchers choose to use for synthesizing. For instance, the mixture of GA and Particle Swarm Optimization (PSO) has led to many contributions. Settles and Soule (2005) propose a general metaheuristic with the combination of both. Rabbani et al. (2013) hybridize Genetic Algorithm with Particle Swarm Optimization (PSO) to tackle the convoluted Inventory Routing Problem (IRP). The same hybridization helped Younes and Benhamida (2011) deal with the Economic Load Dispatch (ELD) problem. Genetic Algorithm has also been hybridized with Tabu Search (Glover, Kelly, & Laguna, 1995), Simulated Annealing (Yu, Fang, Yao, & Yuan, 2000), and even Neural networks (Yang, Wu, Jin, & Xu, 2016).

One other important aspect of Genetic Algorithm research focuses on finding the best values for the behavioral parameters. The performance of GA is very dependent on its parameters such as population size, crossover, and mutation rates. In the literature there are two sides to this matter. First, some works purely study GA to understand its behavior with different parameters for different problems (Gibbs, Maier, & Dandy, 2015). Also, there are studies that fine tune a proposed algorithm for their own problem, and Tagochi is famously used for that purpose (Ho, Tsai, Lin, & Chou, 2009; Tsai, Liu, & Chou, 2004).

1.2. Contributions

Despite the existence of several great advances making the algorithm better suited for different types of problems, such as multi-objective, distributed decision making problems, leveraging GA's dexterity to approach different problems, rarely has there been any study focusing on improving GA's essence itself. This paper, motivated by an effort to bring Genetic Algorithm closer to its biological foundations, is altering some of its existing parts and adding new features and concepts to enhance its capabilities. Through experimenting, the study shows that Fluid Genetic Algorithm is faster, more accurate, more reliable and more adaptable than the standard GA.

1.3. Paper preface

The remainder of the paper is organized as follows: Section 2 will introduce the classic GA and prepare the reader for Section 3 which presents Fluid GA and the difference that exists between the two. Following the scientific conventions, the last three sections are experiments and results, discussions, and conclusion and future research.

2. Genetic algorithm

Genetic Algorithm, similar to many other meta-heuristics, is an evolutionary population based algorithm. That is to say, a population of answers will evolve through the course of the optimization to move toward the optimality of a problem. Answers or individuals in GA are presented in chromosomes, which, incidentally, are the very strong suit of the algorithm. A chromosome is in fact an answer to the problem which is encoded to form a chromosome. The most prevalently applied chromosome is the binary chromosome. Each GA, consequently, needs to have a decoding function with the purpose of converting chromosome encoding to answers.

Fig. 1 presents a general flowchart of the algorithm. GA will initialize by randomly producing chromosomes as many as the number of populations. In the case of binary chromosome, the cells of the chromosome will be filled with 0 or 1 by the same chance. Each and every chromosome will be decoded to an answer and consequently, their fitness value will be calculated. Fitness value, by definition, is the goodness of the answer according to the problem. Next, the population will be sorted based on the individuals' fitness value. Crossover and mutation are the two very important operators of the algorithm. In both, a selection function plays an essential role. Basically, the operators will change entering chromosomes with the hope of improving them, but choosing which chromosome to undergo the operations is by the selection function. The function that is used for this purpose time and again is the roulette wheel function. This function operates in a way such that each and every member of the population has a chance to be selected, but the better the fitness value of a chromosome the more selection chance it will have.

Crossover is a bi-chromosomal operator in the sense that it will work on two chromosomes to output other(s). Two entering chromosomes will mix and produce one or two new chromosome(s) which are known by their offspring. In the case of the binary chromosome, one-cut crossover is most used. In one-cut crossover, both chromosomes will be broken from the same cell number and their parts will swap between the two, resulting in two different chromosomes that have characteristics of both entering chromosomes. Crossover is famous for being GA's optimality derive, swaying the population toward best answers.

Mutation, unlike crossover, is not bi-chromosomal and does not serve the purpose of moving the population toward optimality. Its contribution to the algorithm is to keep it from local optima by radically changing the entering chromosomes. The single entering chromosome is changed by the operator harshly, without any reason, and randomly. Last word about mutation is the extent that operator will change the entering chromosome. Mutation rate is the term for this behavioral factor of the algorithm.

The pivotal step in the algorithm and certainly in the flowchart is deciding when to stop the evolution and be satisfied with the best answer in hand. There is no way GA can be sure of the optimal solution unless an optima is known to it in advance so there is a need for stopping strategies. In fact, there are different stoppage criteria. They can be as simple as a specific numbers of iterations or more involved by bringing the scaled improvements into equation.

3. Fluid Genetic Algorithm (FGA)

3.1. FGA and GA distinction

Fluid Genetic Algorithm (FGA) is in fact a Genetic Algorithm with some fundamental differences. These differences are