Aerospace Science and Technology 14 (2010) 56–64

Contents lists available at ScienceDirect

Aerospace Science and Technology

www.elsevier.com/locate/aescte

Vibrational genetic algorithm enhanced with fuzzy logic and neural networks

Y. Volkan Pehlivanoglu [a,*], Oktay Baysal [b]

[a] Air Force Academy, ASTIN, 34149, Istanbul, Turkey
[b] Old Dominion University, Batten College of Engineering and Technology, Norfolk, VA 23529, USA

ARTICLE INFO

Article history:
Received 30 May 2008
Received in revised form 4 August 2009
Accepted 4 November 2009
Available online 10 November 2009

Keywords:
Aerodynamic design optimization
Fuzzy logic
Genetic algorithms
Neural networks

ABSTRACT

A new optimization algorithm called multi-frequency vibrational genetic algorithm (mVGA) is significantly improved and tested for two different test cases: an inverse design of an airfoil in subsonic flow and a direct shape optimization of an airfoil in transonic flow. The algorithm emphasizes a new mutation application strategy and diversity variety, such as, the global random diversity and the local controlled diversity. The local controlled diversity is based on either a fuzzy logic controller or an artificial neural network depending on the problem type. For both of the demonstration problems considered, remarkable reductions in the computational times have been accomplished.

1. Introduction

Typical aerodynamic design problems are strongly nonlinear and they often exhibit discontinuous derivatives for the objective as well as the constraint functions. Over the years, a diverse array of optimization techniques have been developed and applied to these problems. Among the successful ones are those that employ the genetic algorithms (GA). Although genetic algorithms can be robust and can exhibit high fidelity, they may often be computationally expensive, as is the case in aerodynamic problems, since each evaluation of the cost function requires intensive computations. Currently there are many genetic operators proposed in the literature [3,4,7]. However, it is difficult to derive any guidelines on which method should be used for which type of problem. Therefore, there are still opportunities to develop new and improved operators. The crossover and mutation operators are essential to the genetic algorithms. A crossover enables the algorithm to extract the best genes from different individuals and recombine them into potentially superior children. A mutation adds to the diversity of a population and it is one of the most important factors that determine the performance of a genetic algorithm. The diversity concept can also be divided into global diversity and local diversity in terms of its search frame. On the other hand, it also can be classified into random diversity and controlled diversity in terms of a search direction. Apart from designing new mutation operators, researchers have put relatively less effort in investigating how to apply mutation operators during the process and what kind of diversity should be provided within the population. In classical mutation operations, the provided diversity is mainly random search based and has global character. Such an approach can easily reduce the efficiency of mutational operations and result in deficient diversity. In their application to optimization problems, traditional mutations appear to have been applied in a random manner; hence the realized improvements appear to have been serendipitous.

The present paper introduces the application of a new multi-frequency vibrational genetic algorithm (mVGA) to speed up the optimization algorithm and overcome such problems as deficient diversity and premature convergence during the optimization. The principal role of this multi-frequency approach is to answer the question of which individuals should be mutated and when they should be mutated. Then, depending on the nature of the problem at hand, the present approach employs fuzzy logic or neural network concepts to provide local but controlled diversity within the population in addition to random global diversity.

To demonstrate, mVGA and its variants are applied to two different test cases. First, a new fuzzy-logic-coupled mVGA is tested on an inverse design problem at low flow speed conditions. Secondly, a new neural-network-coupled mVGA is tested on an airfoil shaping problem at transonic flow conditions that mitigates the adverse shock wave effects. Based on the results obtained, it is concluded that the variants of the present multi-frequency vibrational genetic algorithm are efficient and fast genetic algorithms since they can successfully avoid all local optima.

2. Methodology

The present multi-frequency, vibrational, genetic algorithm (mVGA) is an improvement of the vibrational genetic algorithm

* Corresponding author. Fax: +90 212 663 2490.
E-mail addresses: v.pehlivan@hho.edu.tr (Y.V. Pehlivanoglu), obaysal@odu.edu (O. Baysal).

1270-9638/$ – see front matter
doi:10.1016/j.ast.2009.11.001

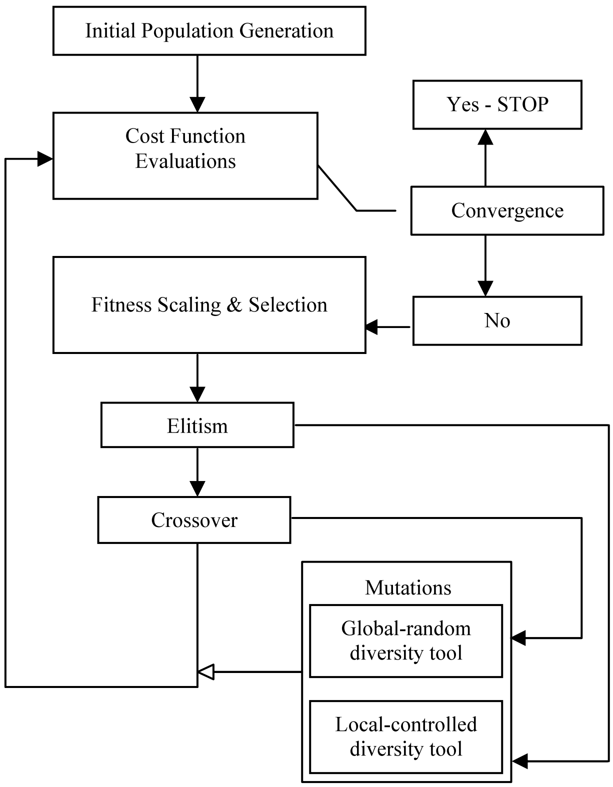

Fig. 1. Flow chart of mVGA algorithm.

(VGA), which is described in [7] and applied in [10,13]. It is an iterative algorithm for which a flow chart is presented in Fig. 1.

An initial population is generated by using a random number operator based on baseline shape or parameters. To describe the method mathematically, let S be the population size, D be the individual (or chromosome) dimension space, f be the objective function, and $\mathbf{Z}_i$ be the current position vector including genes, $z_{i,j}(t)$, described in tth iteration:

$$\mathbf{Z}_i(t) = \left(z_{i,1}(t), z_{i,2}(t), \ldots, z_{i,D}(t)\right)$$
$$z_{i,j}(t) \in R^D, \quad i = 1, 2, \ldots, S \tag{1}$$

The second step is to evaluate the fitness of the current population via a defined cost function f. Then, the cost weighting fitness scaling and roulette selection procedure [8] for mating are determined. The elitism concept is applied next to ensure that the best objective function value within a population is not reduced from one generation to the next. The procedure for the elite fitness value, f^e, and elite individual, $\mathbf{Z}^e$, is as follows:

$$f^e(t) = \underset{\mathbf{Z}_i(t)}{\arg\min}\, f\left(\mathbf{Z}_i(t)\right) \quad \text{and} \quad \mathbf{Z}^e(t) = \mathbf{Z}_i(t) \tag{2}$$

$$\mathbf{Z}^e(t) = \begin{Bmatrix} \mathbf{Z}^e(t-1), & \text{if } f^e(t) > f^e(t-1) \\ \mathbf{Z}^e(t), & \text{if } f^e(t) \leqslant f^e(t-1) \end{Bmatrix} \tag{3}$$

The crossover technique denoted by BLX-α and described in [5] with $\alpha = 0.5$, is applied for the new individuals. The present mVGA mutation strategy is applied right after this crossover phase. At this step, there are two tools. As the first tool, the goal of the first mutation application is to provide a global random diversity in the population. For this reason, all the genes in all the chromosomes are mutated as follows:

$$z_{i,j}(t) = \begin{Bmatrix} z_{i,j}(t)[1 + w_1\beta_1(1-u)]^{i=1,2,\ldots,S}_{j=1,2,\ldots,D}, \\ \quad \text{if } t = nf_1,\ n = 1, 2, \ldots \\ z_{i,j}(t), \quad \text{if } t \neq nf_1,\ n = 1, 2, \ldots \end{Bmatrix} \tag{4}$$

where f_1 is the application frequency, β_1 is a user defined amplitude parameter, u is a random real number between 0 and 1, and w_1 is a user defined scale factor. Implementing the mutation starts from the first gene position of the first chromosome, and continues throughout the genes at the same positions in the other chromosomes. As a second tool, the goal of the second mutation application is to provide a local but controlled diversity in the population. A fuzzy logic controller or a neural network application can be used at this stage depending on the problem at hand. For example, in an inverse design problem, the target is defined in the beginning. Therefore, using fuzzy logic is proper. However, in a direct optimization problem, such a target is not provided. Hence, a neural network application can be used to provide a local-controlled diversity within the population. In applying the fuzzy logic, modified elite genes, $z^{\text{m.e.}}$, are generated as given below:

$$z_s^{\text{m.e.}}(t) = \begin{Bmatrix} z_s^e(t) + w_2\beta_2, & \text{if } t = nf_2,\ n = 1, 2, \ldots \\ z_s^e(t), & \text{if } t \neq nf_2,\ n = 1, 2, \ldots \end{Bmatrix} \tag{5}$$

$$\beta_2 = \mathscr{f}_{\text{func}}\left(\mathbf{Z}^e\right) \tag{6}$$

where f_2 is the application frequency, β_2 is the amplitude, w_2 is a user-defined scale factor, and s is a randomly determined gene number. Instead of fixing the value of β_2, it is estimated by the fuzzy logic controller function, $\mathscr{f}_{\text{func}}$. The modified elite gene is placed in an elite individual and this new individual is randomly located within the population as follows and applied I times:

$$Z^{\text{m.e.}}(t) = \left(z_1^e(t), z_2^e(t), \ldots, z_s^{\text{m.e.}}, \ldots, z_D^e(t)\right) \tag{7}$$

$$\left(Z_k(t)\right)_j = Z^{\text{m.e.}}(t)\Big|^{k=\text{rand}[1-D]}_{j=1,2,\ldots,I} \tag{8}$$

In the neural network application, all the genes of an elite individual are mutated as follows:

$$P_{i,j}(t) = \begin{Bmatrix} z_j^e(t)[1 + w_2\beta_2(1-u)], \\ \quad \text{if } t = nf_2,\ n = 1, 2, \ldots \\ z_j^e(t), \quad \text{if } t \neq nf_2,\ n = 1, 2, \ldots \end{Bmatrix}^{i=1,2,\ldots,N}_{j=1,2,\ldots,D} \tag{9}$$

where u is a random real number between 0 and 1, β_2 is a user-defined constant amplitude. A newly generated temporal population $\mathbf{P}$ includes N individuals. The objective function values of this population, $\mathbf{f}^{NN}$, are predicted via trained neural network function, $\mathcal{N}_{\text{func}}$, and the best I of them are randomly placed within the population:

$$\mathbf{f}^{NN} = \mathcal{N}_{\text{func}}(\mathbf{P})$$
$$\left[\mathbf{f}^{NN}\ \text{order}\right] = \text{sort}\left(\mathbf{f}^{NN}\right)$$
$$\left(Z_k(t)\right)_i = \mathbf{P}_{\text{order}(i)}\Big|^{k=\text{rand}[1-D]}_{i=1,2,\ldots,I} \tag{10}$$

The frequencies f_1, f_2, and I are user-defined constants and their typical values are 5, 2, and 3, respectively. The special functions $\mathscr{f}_{\text{func}}$ and $\mathcal{N}_{\text{func}}$ will be explained in detail when presenting the case studies. After the mutation operations, a new population is evaluated via the cost function which is determined by the real solver. The algorithm repeats all of the above steps as necessary until the convergence criterion are satisfied.

3. Results

The mVGA algorithm will now be applied first to an inverse design problem, followed by a direct shape optimization of an airfoil in transonic flow. The algorithm, however, will be coupled with a fuzzy logic controller for the first application. The lack of a target in a direct optimization problem makes use of fuzzy logic difficult for the second application. However, using neural networks may be appropriate to provide the necessary local but controlled diversity