Expert Systems with Applications 38 (2011) 6254–6261

Contents lists available at ScienceDirect

Expert Systems with Applications

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

Center Based Genetic Algorithm and its application to the stiffness equivalence of the aircraft wing

Jia-qing Zhao [a], Ling Wang [b,*]

[a] Department of Mechanical Engineering, Tsinghua University, Beijing 100084, China
[b] Tsinghua National Laboratory for Information Science and Technology (TNLIST), Department of Automation, Tsinghua University, Beijing 100084, China

ARTICLE INFO

Keywords:
Center Based Genetic Algorithm
Cauchy preferential crossover
Central chaotic mutation
Space shrinking strategy
Stiffness equivalence problem

ABSTRACT

In this paper, a kind of Center Based Genetic Algorithm (CBGA) is proposed to overcome the premature convergence and to solve the stiffness equivalence problem. With the information of population center, central chaotic mutation and space shrinking strategy are designed to guide the evolutionary searching process. Meanwhile, the rank value based roulette selection and a new Cauchy preferential crossover operator are collaboratively used with the mutation operators in the CBGA. To avoid the loss of the best solution, the elitist strategy is employed as well. In addition, local search is embedded in the CBGA after the main evolutionary search process to enhance the exploitation ability by further improving the elite solution. Numerical simulation and comparison based on a set of benchmark functions show that, the proposed CBGA is superior to the hybrid genetic algorithm from the literature in terms of searching efficiency, solution quality, robustness and success ratio. Finally, CBGA is applied to successfully solve the stiffness equivalence problem of the small-aspect-ratio aircraft wing to tapered beam.

1. Introduction

As one of the key branches of evolutionary computation, genetic algorithm (GA) has been widely applied to solve optimization problems in a variety of fields due to its generality and robustness (Holland, 1992). However, it has been shown that simple GA is often easy to converge prematurely. How to overcome the premature convergence has always been an important issue in designing the evolutionary algorithms (Andre, Siarry, & Dognon, 2001; Bilbro & Snyder, 1991; Chelouah & Siarry, 2000; Hrstka & Kuerová, 2004; Hsieh, Sun, & Liu, 2009; Liu, Cai, & Liu, 2000; Tutkun, 2009).

The premature convergence can be regarded that the population stops evolving towards better solutions, thus the algorithm cannot obtain the global optimum (Xiong & Zhao, 2001). Usually, premature convergence is resulted by the loss of diversity when the individuals in the population are very similar to each other. Population size, selection pressure, mutation rate, fitness function property and population initialization are the main factors to affect the premature convergence (Zhou, Yuan, & Zhang, 2007). So far, researchers have proposed many modified genetic algorithms to overcome the premature convergence. The Niched GA was proposed by introducing pre-selection mechanism into GA (Cavicchio, 1970, 1972). De Jong (1975) showed that the larger the mutation rate is, the more GA is apt to random search. That is, large mutation rate will deteriorate the convergence property. To enhance the searching efficiency and enrich the population diversity, adaptive mutation rate and multi-point crossover operators were employed (Leite & Topping, 1998). Theoretically, the concept of *degree of population diversity* was introduced, and *Markov chain* was used to analyze the premature convergence (Leung, Gao, & Xu, 1997). In addition, some strategies by refining or changing the searching space are also used to overcome the premature convergence problem. A scale factor was used in calculating the crossover probability and searching intervals were limited (Andre et al., 2001).

Most existing improved work attempts to avoid premature convergence by increasing population variety, varying crossover and mutation rates, and modifying the genetic operators. By observing the searching behavior of GA, we find that to some extent the center of population always moves towards the global optimization while the individuals move randomly. In addition, the migration scope of the center is smaller than that of the individuals, and even without the mutation operator the population center is prone to move forward to the global optimum. If suitable genetic operators are designed and proper search space shrinking methods are used, the center may migrate to the global optimum faster. With this motivation, we will propose a kind of Center Based Genetic Algorithm (CBGA) in this paper to overcome premature convergence by using the center to guide the evolution process and introducing a new Cauchy preferential crossover operator as well as central chaotic mutation operator and space shrinking strategy. Numerical simulation results and comparisons with an existing hybrid GA

* Corresponding author. Tel.: +86 10 62783125; fax: +86 10 62786911.
E-mail address: wangling@tsinghua.edu.cn (L. Wang).

0957-4174/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2010.11.106

demonstrate the effectiveness of the proposed CBGA. Furthermore, the CBGA is applied to solve the stiffness equivalence problem of the small-aspect-ratio aircraft wing to tapered beam successfully.

The rest of the paper is organized as follows. In Section 2, the CBGA is proposed and explained in detail. To better understand the algorithm, an example is used to illustrate the behavior of the CBGA in Section 3. Numerical simulation results and the comparisons are provided in Section 4 based on 21 benchmark functions. In Section 5, the CBGA is used to solve the stiffness equivalence problem of the wing to the tapered beam, and the comparison with the standard GA is given as well. Finally, we end the paper with some conclusions in Section 6.

2. Center Based Genetic Algorithm (CBGA)

Considering the following unconstrained continuous optimization problem:

$$\min f(x) \quad s.t.\ x \in S \tag{1}$$

where, f is the objective function, $x \in R^n$ denotes searching solution, $S = \prod_{i=1}^{n}[lb_i, ub_i]$ denotes the n-dimensional searching space, lb_i and ub_i denotes the lower bound and upper bound of the ith dimension respectively.

Based on real value encoding, the CBGA uses the rank value based roulette selection, a new Cauchy preferential crossover operator, two new mutation operators and a local search to enhance the optimization ability. Next, we will introduce the main elements of the CBGA.

2.1. Fitness value and selection operation

Denote the population X as $\{x_j = [x_{j,1}, \ldots, x_{j,n}],\ j = 1, \ldots, P\}$, where P denotes the population size and each individual is an n-dimensional real value vector.

Similar to most existing genetic algorithms, a population of initial individuals in the CBGA are also generated randomly in the searching space S. Different from the traditional GA, the CBGA uses rank based fitness function. In particular, the fitness value $fit(x_j)$ of individual x_j is defined as follows. First, sort all the individuals according to their function values in a descending order; Then, assign the rank value as fitness value $fit(x_j)$. That is, $fit(x_j)$ is between P and 1. The smaller function value is, the larger fitness value is.

Selection operation is performed based on the roulette-wheel scheme according to the fitness values. The probability $p(x_j)$ to select x_j is as follows:

$$p(x_j) = 2fit(x_j)/[P(P+1)] \tag{2}$$

2.2. Definition of the population center

The center of population PC_g before the gth generation in the CBGA is defined as follows:

$$PC_g = \frac{2}{P(P+1)} \sum_{j=1}^{P} x_j \cdot fit(x_j) \tag{3}$$

It can be seen that the population center is the weighted center of all individuals' positions, where the fitness values as used as weights. Clearly, the center might be varying at different generations. In CBGA, the center will be used to guide the evolving process.

2.3. Cauchy preferential crossover

When performing crossover and mutation operators, all the individuals x_i in the population X will be normalized to x'_i in population X' from space $[lb_l, ub_l]^n$ to $[0, 1]^n$ by $x'_{i,l} = lb_l + (ub_l - lb_l) \cdot x_{i,l}$, $l = 1, \ldots, n$.

Let C_r be the crossover rate, the Cauchy preferential crossover operator is defined as follows.

First, for individual x'_i generate a random value R_n that is uniformly distributed between 0 and 1. If $R_n > C_r$, then x'_i will not be selected and the next individual x'_{i+1} will be judged; else if $R_n \leqslant C_r$, x'_i will perform the crossover operator.

Next, select one individual x'_j from the rest $P-1$ individuals (population excluding x'_i) by using roulette-wheel selection.

Then, a new individual x^{new} will be produced with the kth component x_k^{new} ($k = 1, \ldots, n$) as follows:

$$x_k^{new} = \begin{cases} x'_{i,k} + \mu \cdot (x'_{j,k} - x'_{i,k}), & f(x_i) \leqslant f(x_j) \\ x'_{j,k} + \mu \cdot (x'_{i,k} - x'_{j,k}), & f(x_i) > f(x_j) \end{cases} \tag{4}$$

where, μ is a random value with Cauchy distribution $f_c(v) = \frac{1}{\pi} \cdot \frac{\lambda}{\lambda^2 + v^2}$ (λ is scale parameter). Comparing with Gauss distribution, the two-side tails of Cauchy distribution function is much longer, which is helpful to generate move with large step size so as to jump out of the local region (Yao, Liu, & Lin, 1999).

Suppose N_P new individuals are generated by crossover, and then the best P individuals will be selected as the new population X'' from all the new individuals and the P parents to further perform mutation operator.

2.4. Central chaotic mutation and population recombination

To enhance the ability of escaping from the local minima, two mutation operators are designed in the CBGA based on the information of the center, that is, central chaotic mutation and population recombination strategy.

Chaos is a kind of nonlinear dynamic that can be used for global optimization due to its ergodicity and random nature. Searching with chaotic variables is often superior to random search (Liu, Wang, Tang, & Huang, 2005; Pan, Wang, & Liu, 2008; Wang, Zheng, & Ling, 2001; Yao, Mei, & Peng, 2002; Yao, Mei, Peng, Hu, & Hu, 2001). In the CBGA, some selected individuals in X'' will perform mutation by adding certain chaotic variables to the center of population, where the Logistic map is used to generate chaotic variables. The central chaotic mutation is implemented as follows:

(1) Select NM individuals that will perform mutation with fitness values between 1 to $P - M_a$ from X'' by roulette-wheel selection, where $NM = \lfloor P \cdot M_r \rfloor$, M_r is the mutation rate, and M_a is the number of best individuals that do not perform central chaotic mutation;
(2) Generate a random vector c_1 in $[0, 1]^n$, and then obtain other CL-1 chaotic vectors with iteration $c_k = 4c_{k-1}(1 - c_{k-1})$, $k = 2, \ldots, CL$, where $CL = \lfloor M_r \times N \times P \rfloor$ and N is the chaos length factor;
(3) Map all the chaotic vectors to search space, and generate chaotic individuals as follows:

$$xc''_k = PC'_g + \xi \cdot (c_k - 0.5), \quad k = 1, \ldots, CL \tag{5}$$

where PC'_g is the normalized population center and ξ is a factor to control the chaos scope.
(4) Evaluate the chaotic individuals, and then determine the best NM ones to replace all the individuals that perform mutation.

Apart from central chaotic mutation, central population recombination strategy is performed in the CBGA. Recombination here means regenerate a population with size P randomly in a new searching space to replace the original population. To avoid population over-crowded, central population recombination will be

performed under following conditions: (a) the function value of the elite individual does not change at some "breakpoint" *RD* of the maximum stagnation generation *MSG*; or (b) the distances between half population of individuals and current normalized population center PC_g are less than *MCD* (minimum crowded distance). Here, "breakpoint" *RD* is set as a vector, whose each element is a real number between 0 and 1. The *i*th element RD_i is corresponding to a stagnation generation $\lfloor RD_i \cdot MSG \rfloor$, and the recombination will be executed at every stagnation generation *STG*.

If any of the above two conditions holds, the original searching space will be shrunk into a new narrowed one around the center of the current population. Then, *P* new individuals will be randomly generated to replace the old population. In particular, a factor is determined first at every *STG*:

$$r_f = s_{\min} + (s_{\max} - s_{\min}) \cdot \exp(-STG/MSG) \tag{6}$$

where $s_{\min}$ and $s_{\max}$ are the minimum and maximum space compression ratio, respectively.

Then, the search space is shrunk as follows:

$$lb_{new} = PC_g - r_f \cdot (lb - ub)/2 \tag{7}$$

$$ub_{new} = PC_g + r_f \cdot (lb - ub)/2 \tag{8}$$

Fig. 1 illustrates the change of the searching space before and after recombination as well as the distribution of the new population.

2.5. Local search and stopping criteria

To avoid the loss of good solutions, the best individual (i.e., elite) of every generation is recorded during the evolution. And to further improve the elite, it will perform local search after the genetic evolution process. The CBGA is implemented on the platform Matlab® 2007b, so we apply the function *fmincon* as the local search that is the integration of Sequential Quadratic Programming, Quasi-Newton and line-search, and set a number *MLSFE* to limit the maximum function evaluation number for local search by using *fmincon*.

Before performing local search, three following criteria are set as the stopping condition for genetic search in CBGA:

(1) The maximal number of the objective function evaluation *MFE-MLSFE* is reached;
(2) The maximal number of the generation *MG* is reached;
(3) The maximal number of the stagnation generation *MSG* is reached;

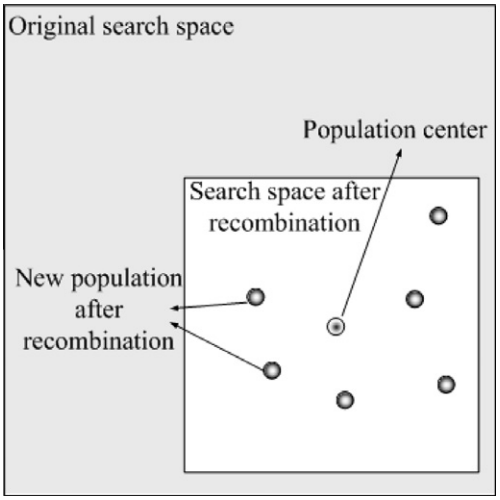

Fig. 1. The new population and search space after recombination.

```
<EVOLUTION>
  Initialization
    Set control parameters
    Population initialization and record elite
  REPEAT
    Calculate population center
    Cauchy preferential crossover and update elite
    Central chaos mutation and update elite
    Recombination mutation if necessary
  Until Stopping criteria are reached
<REFINEMENT>
    Refine elite with fmincon and return
```

Fig. 2. Procedure of CBGA.

2.6. Procedure of CBGA

With the above implementation, the procedure of CBGA is summarized as Fig. 2.

3. Illustration of center's moving in CBGA

To better understand of the idea of the CBGA, we apply the CBGA to minimize the following two-dimensional *Rastrigin* function to illustrate the searching behavior of the algorithm

$$f(x) = \sum_{i=1}^{2}(x_i^2 - 10\cos(2\pi x_i) + 10), \quad -1 \leqslant x_i \leqslant 19 \tag{9}$$

The landscape of the Rastrigin function on $[-1, 19]^2$ is shown in Fig. 3, where the global optimum is $f^*(0, 0) = 0$.

We run the CBGA with population size 10 in two cases: (1) CBGA in Fig. 2 without the central chaotic mutation and population recombination; and (2) CBGA in Fig. 2.

For the case 1, the trace of the population center is shown in Fig. 4. The algorithm executes 37 generations and finally stops at the local optimum [0.995, 0]. From the figure, it can be seen that as the population evolves the population center gradually moves towards the global optimum even though it fails to reach the global minimum, which inspires us to get the following ideas: (i) The population center could be used to guide the evolution of population; (ii) Chaotic disturbance could be added to population center as the

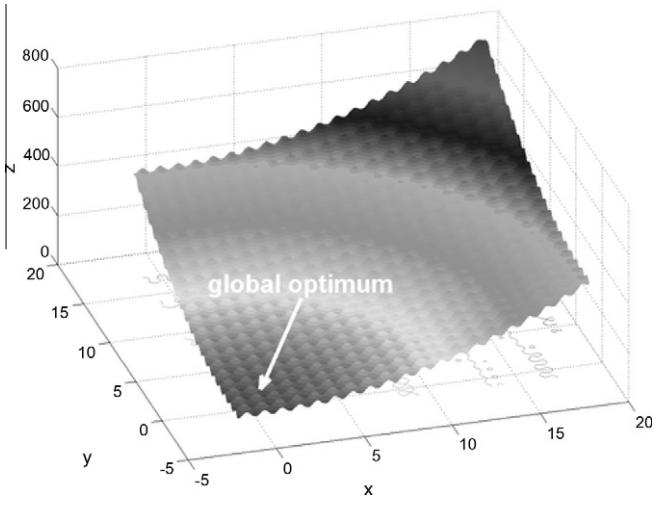

Fig. 3. Landscape of the Rastrigin function.

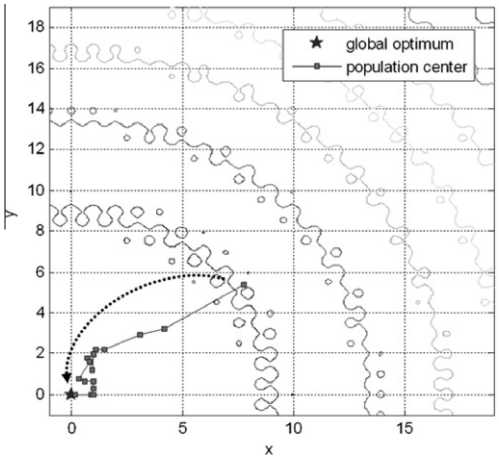

Fig. 4. The trace of population center when only using Cauchy preferential crossover.

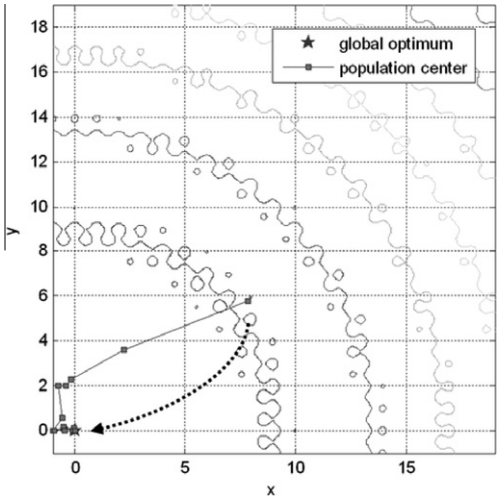

Fig. 5. The trace of population center when using CBGA.

mutation operator to reduce the randomness of traditional mutation operators; (iii) The searching space could be shrunk to a smaller one around population center to improve the convergence; (iv) Shrinking the searching space but conditionally replacing some individuals with random ones could be helpful to enrich the diversity to prevent premature convergence. All the above are just the ideas to design the CBGA in Section 2.

From Fig. 5, the trace of population center when using the CBGA is illustrated. By using the information of center (i.e., central chaotic mutation and population recombination), the population center moves towards the global optimum efficiently, and the elite arrives at the global optimum only with 19 generations. So, it shows the effectiveness of our idea to improve the genetic algorithm.

4. Numerical simulation

Chelouah and Siarry (2003) designed a continuous hybrid algorithm (CHA) by hybridizing the continuous GA and Nelder–Mead simplex search. The CHA was tested with 21 functions listed in Appendix Table A2, and the parameters of some functions are listed in Appendix Tables A1–A4. The CHA utilized the idea of "promising area" which is similar to the space shrinking in CBGA, and it showed that the performance of CHA is similar or superior to six different algorithms. So, we also test the performance of CBGA with these 21 functions and compare it with CHA.

For each function, the algorithm is both run 100 times independently. We summarize the following indexes: the rate of successful minimizations, the average of the objective function evaluation numbers, and the average derivation. The average derivation is defined as the average of $|f_{CBGA} - f_K|$ in 100 runs, where f_{CBGA} is the objective value obtained by CBGA and f_K is the known minimum. Same as CHA (Chelouah & Siarry, 2003), a CBGA run is called a successful run if the following inequality holds:

$$|f_{CBGA} - f_K| < \varepsilon_{real} \cdot |\langle f_e \rangle| + \varepsilon_{abs} \tag{10}$$

where, $\varepsilon_{real} = 1e{-}4$, $\varepsilon_{abs} = 1e{-}6$, and $\langle f_e \rangle$ is an empirical average objective value that is calculated with typically 100 points randomly selected inside the search domain before running the algorithm. For all the successful runs, we will calculate the average objective function evaluation number for comparing the efficiency.

Table 1
Results of CHA and CBGA for 21 benchmark testing functions.

Test function	Success rate (%)		Average function evaluation numbers		Average derivation between the best successful value and the known value	
	CHA	CBGA	CHA	CBGA	CHA	CBGA
RC	**100**	**100**	295	**166**	1.00E−04	**3.76e−07**
ES	**100**	**100**	952	**927**	1.00E−03	**2.86e−08**
GP	**100**	**100**	259	**239**	1.00E−03	**1.10e−08**
B2	**100**	**100**	**132**	325	2.00E−07	**2.61e−09**
SH	**100**	**100**	**345**	533	5.00E−03	**8.83e−06**
R2	**100**	**100**	459	**324**	4.00E−03	**1.30e−07**
Z2	**100**	**100**	215	**121**	3.00E−06	**8.45e−08**
DJ	**100**	**100**	371	**92**	2.00E−04	**1.71e−16**
H3;4	**100**	**100**	492	**485**	5.00E−03	**5.76e−07**
S4;5	**85**	42	698	**562**	9.00E−03	**4.21e−07**
S4;7	**85**	49	620	**559**	1.00E−02	**4.82e−07**
S4;10	**85**	48	635	**558**	1.50E−02	**2.17e−07**
R5	**100**	**100**	**3290**	4450	1.80E−02	**1.73e−07**
Z5	**100**	**100**	950	**732**	6.00E−05	**6.80e−07**
H6;4	**100**	70	930	**884**	8.00E−03	**3.26e−06**
R10	83	**93**	14,563	**5832**	8.00E−03	**1.56e−07**
Z10	**100**	**100**	4291	**2696**	1.00E−06	**3.40e−07**
R50	79	**93**	55,356	**22,798**	5.00E−03	**7.38e−06**
Z50	**100**	**100**	75,520	**56,646**	1.00E−05	**5.18e−06**
R100	72	**89**	124,302	**36,012**	8.00E−03	**4.57e−06**
Z100	**100**	**100**	95,246	**49,151**	1.00E−03	**3.61e−05**

Note: The bold values denote better results of those obtained by CHA and CBGA.

The parameter setting of CBGA is listed in Appendix Table A5, and the results of CBGA and CHA are listed in Table 1.

From Table 1, it can be seen that both CBGA and CHA obtain 100% success rate for eight 2-dimensonal functions, which means the two methods are both effective for low dimensional problems. For 6 out of 8 functions, CBGA uses less function evaluation numbers (FEs) than CHA, and all the derivations by CBGA are less than that by CHA. So, CBGA is more efficient and robust than CHA. For the other 13 functions with 3–100 variables, CBGA obtains best results with smaller derivation than CHA and CBGA costs less FEs for 12 functions than CHA. So, CBGA is still effective for high dimensional functions.

5. Application to stiffness equivalence problem

Wings are important parts of fighting aircrafts and airliners. In addition to supply lifting power, the wing also takes the responsibility to be equipped with the undercarriage, engine and other devices such as auxiliary fuel tank and missiles. Usually, when the plane lands on the airport, we need to know the wing's deformation caused by the gravity of fuel tank or missiles installed under wingtip or other locations. The wing's aspect ratio is defined as the ratio of span to average geometry chord length. The smaller the aspect ratio is, the larger the stiffness is. As for the wing with small-aspect-ratio, it is hard to get large deformation. On the basis of this fact, the specific wing could be equated with one elastoplastic tapered beam on the aspect of the stiffness, if the equivalence could be implemented. The deformation computation of the complicated physical wing that is usually implemented by the finite element method (FEM) (Zienkiewicz, 1989) could be significantly simplified by analytically computing the deflection of the tapered beam.

In this paper, the equivalence is implemented by following steps: First, the true stiffness that is the *F–U* curve of the wing is acquired through finite element method; Then, the deflection of the tapered beam is analyzed based on the small deformation assumption; Finally, we optimize the parameters of the tapered beam by CBGA to make the tapered beam's displacements close to displacements of the *F–U* relationship on the feature points so as to obtain a tapered beam with equal stiffness as the real wing.

5.1. Finite element analysis for the wing's stiffness

A scaled wing model with aspect ratio 3.33 and wingspan size 250 mm is taken as an example to show the process of the stiffness equivalent. The thickness and materials of the wing's skin and rib are list in Table 2.

We get the force and displacement relationship on the wingtip (*F–U*) by the static analysis module of the commercial FEM software ABAQUS®. The nodes on wing root are totally fixed and the concentration force *F* varying from 0 to 800 N is applied on the wingtip. In Fig. 6, it illustrates the boundaries of the wing and Von Mises stress contour obtained by FEM simulation. The resultant *F–U* relationship is shown as the solid line in Fig. 9 in Section 5.3, where *U* is the displacement on the direction of *F*.

Table 2
Material property of the wing.

	Thickness (mm)	Elastic modulus (MPa)	Poisson's ratio	Yield strength (MPa)	Tangent modulus (MPa)
Rib	1.0	68,000	0.32	120	1120
Skin	1.2	70,000	0.27	145	1200

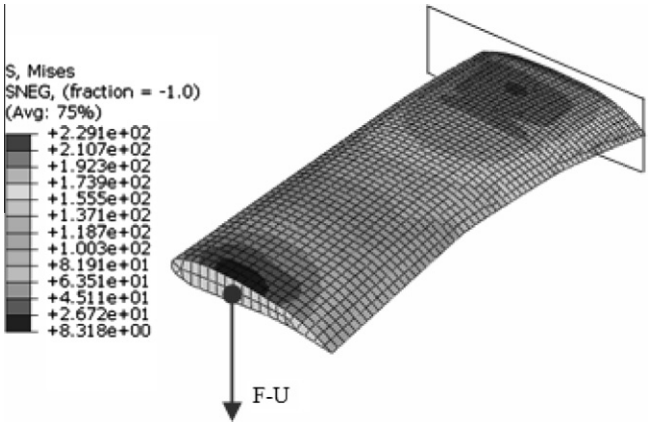

Fig. 6. Boundaries of the wing and Von Mises stress contour obtained by FEM simulation.

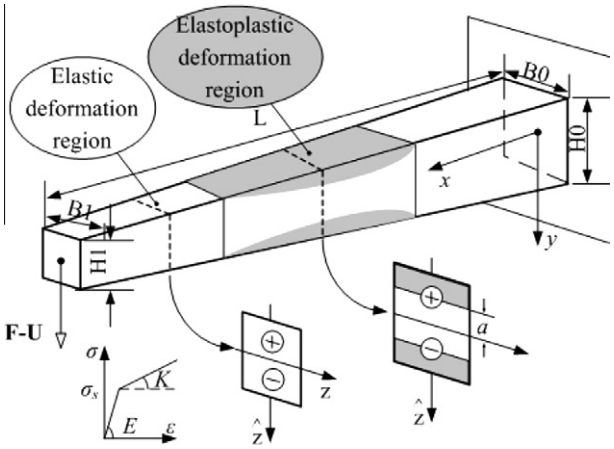

Fig. 7. Seven parameters of tapered beam and its elastic and elastoplastic region.

5.2. Elastoplastic analysis of the tapered beam

For the tapered beam shown in Fig. 7, the bilinear hardening material model is used to describe the constitutive relationship and a transverse force *F* is applied on the free end. Seven parameters concerning the deformation of the tapered beam are listed as follows: section width at the beam's root (*B*0), section height at the beam's root (*H*0), section width at the beam's free end (*B*1), section height at the beam's free end (*H*1), elastic modulus of the material (*E*), tangent modulus of the material (*K*), and yield strength of the material (σ_s).

With the bilinear hardening material model the stain-stress relationship is expressed as:

$$\sigma = \begin{cases} E\varepsilon, & \varepsilon \leqslant \varepsilon_s \\ \sigma_s + K(\varepsilon - \varepsilon_s), & \varepsilon > \varepsilon_s \end{cases} \tag{11}$$

where σ is stress, σ_s is yield stress, ε is strain, E is elastic modulus, K is tangent modulus, and $\varepsilon_s = \sigma_s/E$ is the yield strain.

The global coordinate system is defined as follows: the global origin *O* is at the center of the beam's root, the *x* axis is directing from *O* to the center of the beam's free end, the *y* axis is the upright from *O*, and *z* axis is determined by the right-hand rule. The cross section width and height of the beam at any location is:

$$\begin{cases} B(x) = B1 - \frac{x}{L}(B1 - B0) \\ H(x) = H1 - \frac{x}{L}(H1 - H0) \end{cases} \quad 0 \leqslant x \leqslant L \tag{12}$$

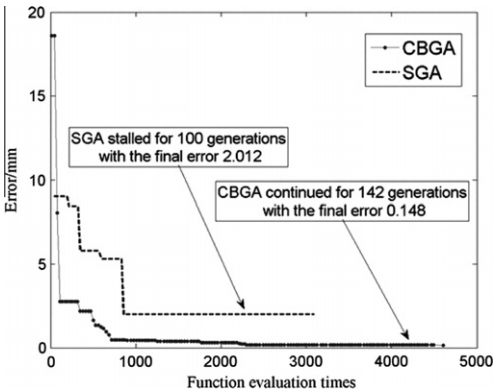

Fig. 8. Evolution processes of CBGA and SGA.

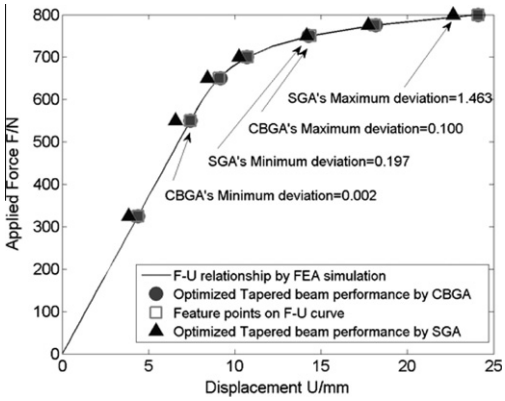

Fig. 9. Beam stiffness obtained by CBGA and SGA.

When the force F on the center of the free end increases to certain value, the beam will subsequently undergo the pure elastic deformation and elastic-plastic deformation. When the elastic-plastic deformation appears, the whole beam will be divided into two parts: elastic deformation region and elastic-plastic region. For the elastic region, the neutral layer deflection on the y direction accords with:

$$\frac{d^2y}{dx^2} = -\frac{F(L-x)}{EI(x)} \tag{13}$$

For the elastic–plastic region, the neutral layer deflection accords with:

$$\frac{d^2y}{dx^2} = \frac{3F(L-x) - 3B(x)(\sigma_s - K\varepsilon_s)(H^2(x)/4 - a^2)}{2B(x)[(E-K)a^3 + KH^3(x)/8]} \tag{14}$$

The points that enter elastic-plastic region accords with:

$$\varepsilon_s = \frac{6F(L-x)}{EB(x)H^2(x)}, \quad 0 \leqslant x \leqslant L \tag{15}$$

On cross section of the elastic-plastic region, the height of elastic area a in Eq. (14) accords with:

$$a^3 + a\cdot\left(\frac{3F}{\varepsilon_s(E-K)}\cdot\frac{L-x}{B(x)} - 3H^2(x)/4\right) + \frac{H^3(x)}{4}\cdot\frac{K}{K-E} = 0 \tag{16}$$

Set $P_v = [B0, H0, B1, H1, E, K, \sigma_s]^T$ as the parameters vector. By solving the ordinary differential equations (13) and (14) with numerical method, we could obtain the relationship between the force applied F on the free end center and the deflection of the free end center on the y direction $Y(F, P_v)$:

$$Y(F, P_v) = y(x = L, F, P_v) \tag{17}$$

where L is the length of the beam and it equals the length of the wing.

5.3. CBGA for parameter optimization of the tapered beam

As shown in Fig. 9, seven points that characterize the feature of the F–U are picked out as the feature points. The feature points are denoted as (F_i, U_i) $(i = 1, 2, \ldots, 7)$. When optimizing, we only calculate the deflection $Y(F_i, P_v)$ caused by seven forces F_i and compare them with U_i to calculate the error. Thus, the optimization problem of equating the wing's stiffness to the tapered beam can be formulated as follows:

$$\min \quad Error(P_v) = \sum_{i=1}^{7}[Y(F_i, P_v) - U_i]^2, \quad LB \leqslant P_v \leqslant UB \tag{18}$$

Given the following empirical values for LB and UB: $LB = [6, 5, 6, 5, 0.1e5, 0.5e3, 100]^T$ and $UB = [20, 10, 40, 30, 10e5, 50e3, 1000]^T$. The parameters of CBGA are set as: $MFE = 5000$, $MG = 150$, $MSG = 50$, $MLSFE = 500$, $P = 15$, $N = 4$, $C_r = 0.9$, $\lambda = 0.1$, $\xi = 0.2$, $MCD = 0.01$, $M_r = 0.3$, $M_a = 5$, $s_{\max} = 0.9$, $s_{\min} = 0.4$, $RD = [0.2, 0.5, 0.8]$. Meanwhile, we use a simple genetic algorithm (SGA) for comparison. In SGA, the uniform arithmetical crossover operator and uniform mutation are employed, and we set the maximum function evaluation 5000, population size 20, crossover rate 0.9, mutation rate 0.2, maximum generations 250, and maximum stagnation generation 100. The evolution processes of CBGA and SGA are shown in Fig. 8.

From Fig. 8, it can be seen that CGBA stops after 142 generations' evolution with 4603 function evaluation times and obtains a solution with error 0.148. In contrast, SGA prematurely converges and stagnates for 100 generations. Although SGA totally costs 3091 function evaluation times, it only obtains a solution with error 2.012. So, it can be concluded that CBGA is of good ability to overcome premature convergence and is of better performances than SGA.

In addition, the parameters of resultant tapered beams obtained by CBGA and SGA are shown in Table 3, and the stiffness of two kinds of beams is shown in Fig. 9. It is clear that the stiffness of the taper beam obtained by CBGA is almost the same as the real wing. By CBGA, the maximum and minimum deflection deviation values under seven F_i are 0.1 and 0.002, respectively, while by SGA the values are 1.463 and 0.197, respectively. So, the proposed CBGA can successfully solve the stiffness equivalence problem of the small-aspect-ratio wing to the tapered beam.

Table 3
The tapered beam parameters optimized by SGA and CBGA.

	$B0$ (mm)	$B1$ (mm)	$H0$ (mm)	$H1$ (mm)	E (MPa)	K (MPa)	σ_s (MPa)
SGA	15.46	9.28	10.16	5.42	590454.28	16472.40	502.39
CBGA	18.79	9.53	7.16	5.97	899448.27	14007.35	766.92

Table A1
Testing functions and their optima.

Name	$f(x)$		Optimal solution
RC	$f(x) = [x_2 - 5/(x\pi^2)]x_1^2 + (5x_1/\pi - 6)^2 + 10[1 - 1/(8\pi)]\cos(x_1) + 10$, $-5 < x_1 < 10, 0 < x_2 < 15$		$x^* = (-\pi, 12.275)$, $(9.42478, 2.475)$ $f(x^*) = 0.397787$
ES	$f(x) = -\cos(x_1)\cos(x_2)\exp\{-[(x_1 - \pi)^2 + (x_2 - \pi)^2]\}$, $-100 < x_j < 100,\ j = 1, 2$		$x^* = (\pi, \pi)$ $f(x^*) = -1$
GP	$f(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]\cdot$ $[30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$, $-2 < x_j < 2,\ j = 1, 2$		$x^* = (0, -1)$ $f(x^*) = 3$
B2	$f(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7$, $-100 < x_j < 100,\ j = 1, 2$		$x^* = (0, 0)$ $f(x^*) = 0$
SH	$f(x) = \sum_{i=1}^{5} i \cdot \cos[(i+1)x_1 + i] \cdot \sum_{i=1}^{5} i \cdot \cos[(i+1)x_2 + i]$, $-10 < x_j < 10,\ j = 1, 2$		$f(x^*) = -186.7309$
Rn	$f(x) = \sum_{j=1}^{n} -[100(x_j^2 - x_{j+1}^2) + (x_j - 1)^2]$, $-5 < x_j < 10,\ j = 1, \ldots, n$		$x^* = (1, \ldots, 1)$ $f(x^*) = 0$
Zn	$f(x) = \sum_{j=1}^{n} x_j^2 + (\sum_{j=1}^{n} 0.5jx_j)^2 + (\sum_{j=1}^{n} 0.5jx_j)^4$, $-5 < x_j < 10,\ j = 1,^*, n$		$x^* = (0, \ldots, 0)$ $f(x^*) = 0$
DJ	$f(x) = x_1^2 + x_2^2 + x_3^2, -5.12 < x_j < 5.12,\ j = 1, 2, 3$		$x^* = (0, 0, 0)$ $f(x^*) = 0$
H3;4	$f(x) = -\sum_{i=1}^{4} c_i \cdot \exp[-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2]$, $0 < x_j < 1,\ j = 1, 2, 3$		$x^* = (0.11, 0.555, 0.855)$ $f(x^*) = -3.86278$
S4;5		$n = 5$	$f(x^*) = -10.1532$
S4;7	$f_{4,n}(x) = -\sum_{i=1}^{n}[(x - a_i)^T(x - a_i) + c_i]^{-1}$, $x = (x_1, \ldots, x_4)^T, a_i = (a_i^1, \ldots, a_i^4)^T, 0 < x_i < 10$	$n = 7$	$f(x^*) = -10.40294$
S4;10		$n = 10$	$f(x^*) = -10.53641$
H6;4	$f(x) = -\sum_{i=1}^{4} c_i \cdot \exp[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2]$, $0 < x_j < 1,\ j = 1, \ldots, 6$		$x^* = (0.20196, 0.150011,$ $0.47687, 0.275332,$ $0.311652, 0.6573)$ $f(x^*) = -3.3223$

Table A2
Coefficients a and c in H3;4.

	a_i^T				c_i
1	4	4	4	4	0.1
2	1	1	1	1	0.2
3	8	8	8	8	0.2
4	6	6	6	6	0.4
5	3	7	3	7	0.4
6	2	9	2	9	0.6
7	5	5	3	3	0.3
8	8	1	8	1	0.7
9	6	2	6	2	0.5
10	7	3.6	7	3.6	0.5

6. Conclusions

This paper developed a Center Based Genetic Algorithm (CBGA) by using the information of the central of population to improve the performances of genetic algorithm. Based on center information, center chaotic mutation and space shrinking strategy were designed to guide the searching process. Meanwhile, a Cauchy preferential crossover was applied in the CBGA as well. Numerical simulation and comparison showed the effectiveness and efficiency as well as robustness of the proposed CBGA. Furthermore, the CBGA was applied to successfully solve the stiffness equivalence problem of the small-aspect-ratio wing to the tapered beam. However, there are some additional parameters to set in CBGA apart from the parameters of classical GA. So, our future work is to develop adaptive algorithms where the parameters are controlled adaptively to alleviate the parameter setting, and apply the algorithm to other engineering design and optimization problems.

Table A4
Coefficient a, c and p in S4;n.

i	a_{ij}			c_i	p_{ij}		
1	3	10	30	1	0.3689	0.117	0.2673
2	0.1	10	35	1.2	0.4699	0.4387	0.747
3	3	10	30	3	0.1091	0.8732	0.5547
4	0.1	10	35	3.2	0.0381	0.5743	0.8828

Acknowledgments

This research is partially supported by National Science Foundation of China (60774082 and 70871065, 60834004) and Program for New Century Excellent Talents in University (NCET-10-0505) and Doctoral Program Foundation of Institutions of Higher Education of China (20100002110014) as well as in part by Siemens Ltd. China (2009-009).

Appendix A

See Tables A1–A5.

Table A3
Coefficient a, c and p in H6;4.

i	a_{ij}						c_i	p_{ij}					
1	10	3	17	3.5	1.7	8	1	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.05	10	17	0.1	8	14	1.2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	3	3.5	1.7	10	17	8	3	0.2348	0.1451	0.3522	0.2883	0.3047	0.665
4	17	8	0.05	10	0.1	14	3.2	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

Table A5
Parameter setting of CBGA for 21 test functions.

	Parameter combination	$[s_{max}, s_{min}]$	RD
RC	[200, 30, 10, 60, 10, 2, 0.9, 0.05, 0.05, 0.05, 0.1, 8]	[0.5, 0.1]	[0.2, 0.5, 0.8]
ES	[950, 20, 10, 40, 20, 4, 1, 0.2, 0.2, 0.01, 0.3, 4]	[0.8, 0.1]	[0.2, 0.5, 0.8, 0.9]
GP	[260, 20, 20, 60, 15, 4, 0.6, 0.1, 0.2, 0.1, 0.3, 10]	[0.8, 0.4]	[0.2, 0.5, 0.8, 0.9]
B2	[350, 25, 10, 60, 10, 2, 1, 0.05, 0.05, 0.05, 0.1, 8]	[0.6, 0.01]	[0.2, 0.5, 0.8, 0.9]
SH	[550, 30, 20, 60, 10, 2, 1, 0.1, 0.25, 0.01, 0.3, 5]	[0.6, 0.2]	[0.2, 0.5]
R2	[400, 10, 10, 150, 20, 8, 0.9, 0.2, 0.1, 0.01, 0.2, 10]	[0.4, 0.1]	[0.5, 0.8, 0.9]
Z2	[150, 50, 10, 50, 5, 5, 1, 0.1, 0.1, 0.1, 0.3, 4]	[0.5, 0.01]	[0.2, 0.5, 0.8, 0.9]
DJ	[100, 10, 10, 20, 5, 4, 0.9, 0.2, 0.2, 0.05, 0.1, 8]	[0.8, 0.1]	[0.2, 0.5, 0.8]
H3;4	[500, 60, 10, 60, 15, 4, 0.8, 0.1, 0.1, 0.1, 0.3, 4]	[0.8, 0.05]	[0.2, 0.5, 0.8, 0.9]
S4;5	[700, 30, 10, 40, 10, 1, 0.9, 0.05, 0.1, 0.1, 0.2, 3]	[0.8, 0.4]	[0.2, 0.5, 0.8]
S4;7	[700, 30, 10, 40, 10, 1, 0.9, 0.05, 0.1, 0.1, 0.2, 3]	[0.8, 0.4]	[0.2, 0.5, 0.8]
S4;10	[700, 30, 10, 40, 10, 1, 0.9, 0.05, 0.1, 0.1, 0.2, 3]	[0.8, 0.4]	[0.2, 0.5, 0.8]
R5	[6000, 60, 40, 300, 100, 4, 0.45, 0.05, 0.1, 0.01, 0.05, 5]	[0.8, 0.5]	[0.5, 0.8]
Z5	[850, 50, 10, 200, 5, 5, 1, 0.1, 0.1, 0.1, 0.3, 4]	[0.5, 0.01]	[0.2, 0.5, 0.8, 0.9]
H6;4	[930, 80, 15, 120, 5, 5, 1, 0.1, 0.1, 0.2, 0.3, 4]	[0.5, 0.1]	[0.2, 0.5, 0.8, 0.9]
R10	[13000, 100, 50, 500, 40, 4, 0.5, 0.1, 0.05, 0.01, 0.2, 10]	[0.9, 0.2]	[0.2, 0.5, 0.8]
Z10	[3000, 250, 10, 700, 5, 5, 1, 0.1, 0.1, 0.1, 0.3, 4]	[0.5, 0.01]	[0.2, 0.5, 0.8, 0.9]
R50	[30000, 350, 60, 6000, 30, 6, 0.5, 0.1, 0.05, 0.1, 0.2, 10]	[0.8, 0.3]	[0.2, 0.5, 0.8, 0.9]
Z50	[60000, 1000, 500, 8000, 40, 15, 1, 0.1, 0.1, 0.1, 0.3, 4]	[0.4, 0.01]	[0.2, 0.5, 0.8, 0.9]
R100	[50000, 250, 60, 30000, 50, 8, 0.4, 0.1, 0.05, 0.1, 0.2, 10]	[0.8, 0.3]	[0.2, 0.5, 0.8, 0.9]
Z100	[90000, 200, 100, 30000, 50, 4, 0.8, 0.1, 0.05, 0.01, 0.2, 10]	[0.9, 0.2]	[0.2, 0.5, 0.8]

Note: Parameter combination = [MFE, MG, MSG, $MLSFE$, P, N, C_r, λ, ξ, MCD, M_r, M_a].

References

Andre, J., Siarry, P., & Dognon, T. (2001). An improvement of the standard genetic algorithm fighting premature convergence in continuous optimization. *Advances in Engineering Software, 32*(1), 49–60.

Bilbro, G., & Snyder, W. (1991). Optimization of functions with many minima. *IEEE Transactions on Systems, Man and Cybernetics, 21*(4), 840–849.

Cavicchio, D. (1970). *Adaptive search using simulated evolution*. Ann Arbor: University of Michigan.

Cavicchio, D. (1972). *Reproductive adaptive plans*. New York: ACM.

Chelouah, R., & Siarry, P. (2000). A continuous genetic algorithm designed for the global optimization of multimodal functions. *Journal of Heuristics, 6*(2), 191–213.

Chelouah, R., & Siarry, P. (2003). Genetic and Nelder–Mead algorithms hybridized for a more accurate global optimization of continuous multiminima functions. *European Journal of Operational Research, 148*(2), 335–348.

De Jong, K. (1975). *Analysis of the behavior of a class of genetic adaptive systems*. Ann Arbor: University of Michigan.

Holland, J. (1992). *Adaptation in natural and artificial systems*. Cambridge: MIT press.

Hrstka, O., & Kuerová, A. (2004). Improvements of real coded genetic algorithms based on differential operators preventing premature convergence. *Advances in Engineering Software, 35*(3–4), 237–246.

Hsieh, S. T., Sun, T. Y., & Liu, C. C. (2009). Potential offspring production strategies: An improved genetic algorithm for global numerical optimization. *Expert Systems with Application, 36*(8), 11088–11098.

Leite, J. P. B., & Topping, B. H. V. (1998). Improved genetic operators for structural engineering optimization. *Advances in Engineering Software, 29*(7–9), 529–562.

Leung, Y., Gao, Y., & Xu, Z. (1997). Degree of population diversity – A perspective on premature convergence in genetic algorithms and its Markov chain analysis. *IEEE Transactions on Neural Networks, 8*(5), 1165–1176.

Liu, J., Cai, Z., & Liu, J. (2000). Premature convergence in genetic algorithm: Analysis and prevention based on chaos operator. In *Proceedings of the 3rd world congress on intelligent control and automation, China* (pp. 387–390).

Liu, B., Wang, L., Tang, F., & Huang, D. X. (2005). Improved particle swarm optimization combined with chaos. *Chaos, Solitons and Fractals, 25*(5), 1261–1271.

Pan, H., Wang, L., & Liu, B. (2008). Chaotic annealing with hypothesis test for function optimization in noisy environment. *Chaos, Solitons and Fractals, 35*(5), 888–894.

Tutkun, N. (2009). Optimization of multimodal continuous functions using a new crossover for the real-coded genetic algorithms. *Expert Systems with Application, 36*(4), 8172–8177.

Wang, L., Zheng, D. Z., & Ling, Q. S. (2001). Survey on chaotic optimization methods. *Computing Technology and Automation, 20*(1), 1–5.

Xiong, W. Q., & Zhao, J. Y. (2001). The premature convergence of genetic algorithms. *Journal of Ningbo University, 10*(2), 23–27.

Yao, X., Liu, Y., & Lin, G. (1999). Evolutionary programming made faster. *IEEE Transactions on Evolutionary Computation, 3*, 82–102.

Yao, J. F., Mei, C., & Peng, X. Q. (2002). The application research of the chaos genetic algorithm (CGA) and its evaluation of optimization efficiency. *Acta Automatic Sinica, 28*(6), 935–942.

Yao, J. F., Mei, C., Peng, X. Q., Hu, Z. K., & Hu, J. (2001). A new optimization approach-chaos genetic algorithm. *Systems Engineering, 19*(1), 70–74.

Zhou, H. W., Yuan, J. H., & Zhang, L. S. (2007). Improved politics of genetic algorithms for premature. *Computer Engineering, 33*(19), 201–203.

Zienkiewicz, O. C. (1989). *The finite element method*. London: McGraw-Hill.