IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 4, Ver. II (Jul-Aug. 2014), PP 54-59
www.iosrjournals.org

Immune Optimization Design of Diesel Engine Valve Spring Based on the Artificial Fish Swarm

Rong Chen[1], Yi Shen[1,2,*], Peng Li[1], Xiaobin Hua[1], Mingxin Yuan[1,2]
[1](*School of Mechanics and Automotive Engineering, Jiangsu University of Science and Technology, Zhangjiagang, China*)
[2](*Suzhou Institute of Technology, Jiangsu University of Science and Technology, Zhangjiagang, China*)

Abstract: *In order to improve the reliability of the structure design of the diesel engine valve spring and its working performance, an artificial fish swarm-based immune genetic algorithm (AFS-IGA) is proposed. After the mutation operation of the immune genetic algorithm (IGA), the train operator and foraging operator of the artificial fish algorithm are added to improve the convergence speed and strengthen the optimal performance of the basic immune genetic algorithm. Compared with genetic algorithm (GA) and the IGA, the function testing results show that the convergence speed of the AFS-IGA is faster, the optimization ability of the AFS-IGA is stronger, and the optimization precision is higher. Furthermore, the optimization design results of the diesel engine valve spring show that the optimization performance of the proposed algorithm is optimal, which verifies the robustness of the AFS-IGA.*

Keywords: *Valve spring, Artificial fish swarm algorithm, Immune genetic algorithm, Optimization design*

I. Introduction

The reasonable design of diesel engine valve spring plays an important role in the improvement of spring working performance, reduction of the engine vibration and noise, and extension of the spring life. At present, many optimization methods are provided for the spring design. Li *et al.* [1] presented the optimization design based on genetic algorithm for the engine valve spring. Although the algorithm is widely used and its parallelism are suitable for solving the large-scale complex optimization problems, the genetic algorithm is easy to fall into local minimum and its convergence speed is slow. Huang *et al.* [2] proposed a design method based on the fuzzy optimization for the internal combustion engine valve spring. The simulation results show that the method is characterized by good robustness; however, its parameters are so many that the inappropriate values will affect the calculation accuracy. Peng *et al.* [3] presented an immune genetic algorithm for the design of clutch compression spring and the optimal design is finished in the MATLAB environment; however, the improvement of the IGA is finished on the basis of the basic genetic algorithm, and the deficiencies regarding the slow convergence speed and low optimization precision still exist. In order to further improve the optimization design effect of the diesel engine valve spring, a new immune genetic algorithm based on the artificial fish is proposed in this paper, and the simulation results verify its validity.

II. Optimization Model of The Valve Spring

In the diesel engine, the reciprocating movement of transmission parts from gas distribution mechanism makes the valve spring bear the alternating load at work [4]. The resonance caused by the cam motion will lead to the vibrating of the valve spring, which will make the amplitude of spring stress increase and the effective spring force decrease. Because the above conditions have a great impact on the diesel engine working, the optimization of the valve spring is particularly important. Due to the limitation of the structure and space, that is to say, each structure size of the spring should be as small as possible, the diameter d of spring wire, mean diameter D of coil, and number n of effective coils are taken as optimization variables, namely:

$$\boldsymbol{X} = \left[x_1, x_2, x_3\right]^T = \left[d, D, n\ \right]^T \tag{1}$$

According to the given work environment and the structure requirements, the objective function is built from the perspective of minimum of the spring weight as follows:

$$\min f\left(\boldsymbol{X}\right) = x_1^2 x_2 \left(x_3 + 1.8\right) \tag{2}$$

The constraints of objective function include: spring index, spring stiffness, spring strength, spring stability, condition of none resonance, minimum number of coils, size of spring diameter, etc. Due to the space limitations, the concrete definitions of above constraint conditions will not be shown here. All definitions can be found in reference [4]. The final constraint conditions after treatment are as follows:

*** Corresponding author.** E-mail: sheny456@hotmail.com

$$g_1(\boldsymbol{X}) = 6.5 - \left| x_2 / x_1 - 9.5 \right| \geq 0 \tag{3}$$

$$g_2(\boldsymbol{X}) = 0.01 - \left| (10^4 x_1^{\ 4} x_2^{-3} x_3^{-1} / 47) - 1 \right| \geq 0 \tag{4}$$

$$g_3(\boldsymbol{X}) = 405 - 2771 x_1^{-2.86} x_2^{0.86} \geq 0 \tag{5}$$

$$g_4(\boldsymbol{X}) = 3.74 - \left[(x_3 + 1.3) x_1 + 15.9 \right] / x_2 \geq 0 \tag{6}$$

$$g_5(\boldsymbol{X}) = 3.56 \times 10^5 x_1 x_2^{-2} x_3^{-1} - 250 \geq 0 \tag{7}$$

$$g_6(\boldsymbol{X}) = x_3 - 3 \geq 0 \tag{8}$$

$$g_7(\boldsymbol{X}) = x_2 - 30 \geq 0 \tag{9}$$

$$g_8(\boldsymbol{X}) = 60 - x_2 \geq 0 \tag{10}$$

$$g_9(\boldsymbol{X}) = x_1 - 2.5 \geq 0 \tag{11}$$

$$g_{10}(\boldsymbol{X}) = 9.5 - x_1 \geq 0 \tag{12}$$

III. Immune Genetic Algorithm

The immune genetic algorithm [5] was presented based on the biological immune mechanism as a kind of improved genetic algorithm [6]. Before the select operation of the basic genetic algorithm, the calculation of antibody concentration based on the information entropy. The higher the concentration of the antibody, the antibody is inhibited more; contrarily the lower the concentration of the antibody, the antibody is accelerated more, which enables the self-regulation of the immune system. In addition, the immune genetic algorithm has memory function, which can speed up its search speed and improve the global search ability.

Supposing that the immune system is composed of *N* antibodies and each antibody has *M* genes, according to the information entropy theory, the information entropy of the j^{th} genes from all *N* antibodies are as follows:

$$H_j(N) = -\sum_{i=1}^{N} P_{ij} \log_2 P_{ij} \tag{13}$$

Where, P_{ij} is the probability that the symbol at j^{th} position is symbol *i* of N antibodies.

The differences among different individuals are described by the average information entropy, and the entropy is defined as:

$$H(N) = \frac{1}{M} \sum_{j=1}^{M} H_j(N) \tag{14}$$

The affinity between antibodies *i* and *j* is defined as:

$$A_{ij} = \frac{1}{1 + H(2)} \tag{15}$$

The affinity between the antibody *i* and antigen *y* is defined as the fitness of the antibody *i*:

$$A_{iy} = J_i(X) \tag{16}$$

Where, $J_i(X)$ is the fitness function of the antibody *i*.

The concentration of antibody *i* is defined as follows:

$$C_i = \frac{1}{N} \sum_{j=1}^{N} Q_{ij} \tag{17}$$

$$Q_{ij} = \begin{cases} 1 & , Q_{ij} > \lambda \\ 0 & else \end{cases} \tag{18}$$

Where, λ is the of threshold value of immune selection and $0.9 \leq \lambda \leq 1$. In this paper, λ = 0.92.

IV. Artificial Fish Swarm Algorithm And Its Basic Operators

4.1 Artificial fish swarm algorithm

Artificial fish swarm algorithm is a kind of swarm intelligence optimization algorithm based on the animal behavior, and it is mainly composed of the prey, swarm and follow operators. The algorithm simulates the foraging behavior of the natural swarm fishes, and makes the population reach the optimal position through

the cooperation among different individuals. A large number of simulation results show that the artificial fish swarm algorithm has strong robustness and good global optimization ability.

4.2 Follow operator

When an artificial fish find the place where have more food and around is not too crowded, other artificial fishes will follow it to find source. Within the perception scope of an artificial fish, it will find the partner who is located in the optimal position and then moves closer to it. If it does not find the optimal partner, it executes the prey operator. The follow operator can speed up the movement of the artificial fish to a better position.

The follow operator can be simply described as follows: Let the current position of artificial fish i be x_i and the food concentration (i.e. fitness value) be y_i, the artificial fish i will search the optimal artificial fish whose position is x_{max} and the fitness value is $y_{\max}$ according to its current position x_i. If $y_{\max}>y_i$, artificial fish i will take $x_{\max}$ as the center to search other artificial fishes within its perception scope. Supposing that the amount of the searched fish is n_f, if $y_{\max}>y_i$ and $y_{\max}/n_f < y_i * \delta$ (δ is the crowded degree factor), the artificial fish i can confirm that the location of $x_{\max}$ has more food and is not too crowded, and will move a step toward the $x_{\max}$ according to Eq. (18). If $y_{\max} < y_i$, the artificial fish i will execute the prey operator.

$$x_{nest} = x_i + rand() \times step \times \left(x_{\max} - x_i\right) / \| x_{\max} - x_i \| \tag{19}$$

4.3 Prey operator

The artificial fish can instinctively find food and swim towards the place where more food has. During the process of searching for the optimal location, the artificial fish finds a better position in its perception scope according to its own position. If the place is not found, the fish will swim randomly. Therefore, the prey operator is a process of finding the optimal solution.

The prey operator can be simply described as follows: Let the current position of artificial fish i be x_i and the food concentration (i.e. fitness value) be y_i. The artificial fish i select randomly a position x_j within its perception scope according to Eq.(20) and calculate the fitness value y_j of the position. If $y_i < y_j$, the artificial fish i moves a step towards the new position according to Eq.(21); on the contrary, if $y_i > y_j$, the artificial fish i randomly select a position x_j again according to Eq.(20) and determine whether the movement condition is met. If the movement condition is still not met after several attempts, the artificial fish i randomly moves a step according to Eq.(22).

$$x_j = x_i + rand() \times visual \tag{20}$$

$$x_{next} = x_i + rand() \times step \times (x_j - x_i) / \| x_j - x_i \| \tag{21}$$

$$x_{next} = x_i + rand() \times step \tag{22}$$

4.4 Immune Genetic Algorithm Based on Artificial Fish Swarm

(1) Initialize the algorithm parameters.

(2) Generate the initial population.

The initial population is often randomly generated in the solution space. If the antigen is a new one, N new antibodies are randomly generated to form initial population *An*, and the memory cell bank is empty. If the antigen has ever occurred, the initial population *An* consists of *M* memory cells and (*N*-*M*) antibodies which are generated randomly.

(3) Calculate the affinity of each antibody according to Eqs. (13), (14) and (15).

(4) Calculate the fitness of each antibody according to the type (16).

(5) Execute the crossover and mutation operations.

(6) According to whether artificial fish found the partner who was in the optimal place, and choose to perform train operator or foraging operator.

(7) Update the optimal solution.

(8) Determine whether the end condition is met. If not, return to Step (3); otherwise, output the optimal solution and end.

V. Experiment Results and Analysis

5.1 Function optimization

In order to verify the optimization performance of the proposed AFS-IGA, six functions, namely, Six-hump camel back function f_1, Branins's rcos function f_2, Goldstein-Price function f_3, Himmelblau's function f_4, Rastrigin's function f_5, Bohachevsky function f_6, are tested using Matlab7.01 on PIV 2.99 GHz 2 GB memory

computer, and the test results are compared with the testing results of GA and IGA. Considering the randomness of three intelligent algorithms (i.e. AFS-IGA, GA and IGA), each function is tested independently for 50 times.

1) Six-hump camel back function f_1

$$f_1(x,y)=\left(4-2.1x^2+y^{4/3}\right)x^2+xy+\left(-4+y^2\right)y^2 \tag{23}$$

Where, $x\in[-5.12,5.12]$ and $y\in[-5.12,5.12]$.

2) Branins's rcos function f_2

$$f_2(x,y)=(y-5.1/(4\times\pi^2)\times x^2+5/\pi\times x-6)^2+10\times(1-1/(8\times\pi))\times\cos(x)+10 \tag{24}$$

Where, $x\in[-5,10]$ and $y\in[0,15]$.

3) Goldstein-Price's function f_3

$$f_3(x,y)=(1+(x+y+1)^2\cdot(19-14x+3x^2-14y+6xy+3\cdot y^2))$$
$$\cdot(30+(2x-3y))^2\cdot(18-32x+12x^2+48y-36xy+27y^2) \tag{25}$$

Where, $x\in[-2,2]$ and $y\in[-2,2]$.

4) Himmelblau's function f_4

$$f_4(x,y)=\left(x^2+y-11\right)^2+\left(x+y^2-7\right)^2 \tag{26}$$

Where, $x\in[-6,6]$ and $y\in[-6,6]$.

5) Rastrigin's function f_5

$$f_5(x,y)=20+x^2-10\cos\left(2\pi x\right)+y^2-10\cos\left(2\pi y\right) \tag{27}$$

Where, $x\in[-5.12,5.12]$ and $y\in[-5.12,5.12]$.

6) Bohachevsky function f_6

$$f_6(x,y)=x^2+2y^2-0.3\cos\left(3\pi x\right)-0.4\cos\left(4\pi y\right)+0.7 \tag{28}$$

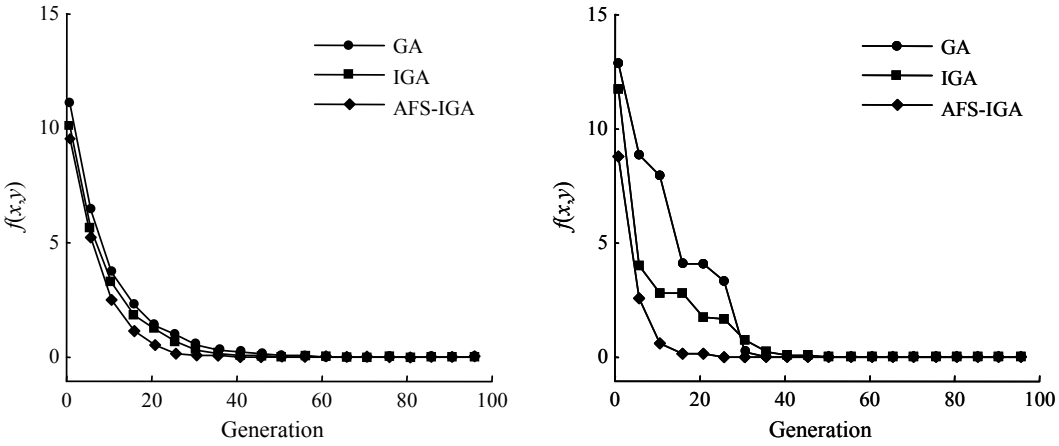
Where, $x\in[-1,1]$ and $y\in[-1,1]$.

From table 1, it can be seen that the global optimization ability of GA is poorer and the convergence speed is slow. Because IGA has the memory function, the search ability of GA is improved and the search speed is accelerated; however, compared with AFS-IGA, the advantage is not obvious. Because the AFS-IGA contains the follow operator and the prey operator, it can quickly find the optimal solution, and its convergence is more stable.

Table 1 Function test results of GA, IGA and AFS-IGA

Function	Precision	Number of the optimal solution			Maximum convergence generation			Average convergence generation			Standard deviation		
		GA	IGA	AFS-IGA	GA	IGA	AFS-IGA	GA	IGA	AFS-IGA	GA	IGA	AFS-IGA
f_1	10^{-3}	31	33	46	886	970	865	311.16	300.79	136.35	250.45	216.54	194.88
f_2	10^{-3}	45	42	47	76	100	674	33.42	32.26	43.77	17.51	23.79	97.73
f_3	10^{-2}	50	50	50	76	67	59	28.60	23.60	18.72	14.96	11.34	9.31
f_4	10^{-3}	42	41	46	835	680	640	97.21	82.90	67.04	179.18	161.52	112.11
f_5	10^{-2}	50	50	50	263	217	177	59.18	54.56	46.46	49.24	38.37	38.52
f_6	10^{-3}	48	46	50	978	935	321	305.96	315.20	63.74	267.60	271.50	78.78

Fig.1 shows the evolutionary curves of GA, IGA and AFS-IGA three algorithms during the function optimization. From the figure, it can be seen that compared with GA, IGA, the convergence speed of AFS-IGA is faster and the global optimization ability is stronger, which further verify the effectiveness of the AFS-IGA.



(a)The average solution (b) The optimal solution

Fig. 1 Average evolution curves of Himmelblau's function

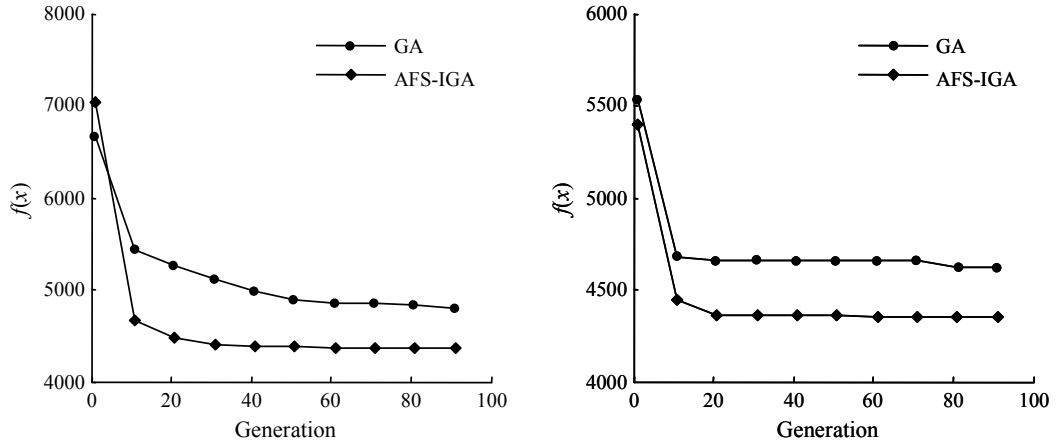
5.2 Structure optimization of the valve spring

In order to further verify the effectiveness and superiority of in the structural optimization of the valve spring, we continue to have a test on the structure optimization. Considering the randomness of the intelligent algorithms, each test of the structure optimization is executed independently for 20 times. The final test results are shown in table 2. The values of the diameter of spring wire, mean diameter of coil, and number of effective coils are rounded up. From table 2, it can be seen that the optimal solution and the mean solution of the proposed AFS-IGA are better than the results of GA, which is mainly because the proposed AFS-IGA contains the follow operator and the prey operator.

Table 2 Comparison of test results between GA and AFS-IGA

Algorithm	GA		AFS-IGA	
	Optimal solution	Average solution	Optimal solution	Average solution
Min$f(\boldsymbol{X})$	4494.8	4821.9	4350.2	4356.8
Number n of effective coils	4	/	4	/
Mean diameter D of coil / *mm*	30	/	30	/
Diameter d of spring wire / *mm*	5	/	5	/

Fig.2 shows the evolutionary curves of GA and AFS-IGA during the structure optimization of the diesel engine valve spring. Through the figure, we can still see that the convergence speed and the optimization ability of the proposed AFS-IGA are better than the results of GA.



(a) Whole population (b) Optimal solution

Fig. 2 Average evolutionary curves of spring optimization

VI. Conclusion

In order to finish the optimization design of the valve spring which has multiple constraints, multiple variables, and bears vibrating load, an immune genetic algorithm based on the artificial fish swarm is proposed. The follow operator speeds up the convergence speed of the proposed AFS-IGA and the prey operator enhances the optimization ability, which makes the AFS-IGA own stronger global optimization ability. Both the function optimization and the diesel engine valve spring optimization verify the validity and superiority of the proposed AFS-IGA.

VII. Acknowledgements

This work is supported by the Modern Educational Technology Project of Jiangsu Province (No. 2013-R-24771), 2013 Teaching Reform and Teaching Study Project from Jiangsu University of Science and Technology (JUST), 2013 Key Teaching Reform and Teaching Study Project from Zhangjiagang campus areas of JUST and 2013 Key Course building Project from Suzhou Institute of Technology.

References

[1] B. Li, M.L. Zhu and X.W. Chen, Optimization design of valve spring of engine based on genetic algorithms, Machine Design, 9, 1998, 22-25.

[2] Q.G. Huan, Design of the internal combustion engine valve spring based on the fuzzy optimization, Science and Technology Innovation Herald, 23, 2012,25-27.

[3] L. Peng, L.L. Zhang and L.N. Yang, Using MATLAB to realize the optimization design of the compressed spring in the clutch based on immune genetic algorithms, Journal of Mechanical Transmission, 1, 2007, 40-42.

[4] Q.S. Gong, Optimized design of 50CrVA steel diesel engine valve spring, Journal of Hunan Institute of Engineering, 13(1), 2003, 30-31,51.

[5] L.A. Chen and P.N. Zhang, Realization of immune genetic algorithm in MATLAB, Journal of Fuzhou University(Natural Science), 32(5),2004, 554-559.

[6] X.F. Wang, X.J. Zhang and X.B. Cao, An improved genetic algorithm based on immune principle, Mini-micro System, 20(2), 1999, 117-120.