An Improved Real-Coded Genetic Algorithm and Its Application

Zhong-Lai Wang, Ping Yang, Dan Ling and Qiang Miao

Abstract—Real-coded genetic algorithm (RGA) usually meets the demand of consecutive space problem. However, compared with simple genetic algorithm (SGA), RGA also has the inherent disadvantages such as prematurity and slow convergence when the solution is close to the optimum solution. This paper presents an improved real-coded genetic algorithm to increase the computation efficiency and avoid prematurity, especially in the optimization of multi-modal function. In this method, mutation operation and crossover operation are improved. Examples are given to demonstrate its computation efficiency and robustness.

Index Terms—Adaptive mutation, arithmetic crossover, elitist strategy, genetic algorithm.

1. Introduction

It has been over 30 years since simple genetic algorithms were put into application. Because genetic algorithms have the inherent advantages as an evolutionary computation, people focus on research to improve its capability. Now genetic algorithm has been successfully applied to many fields, such as biology field\cite{1,2}, combination optimiza-tion\cite{3,4}, engineering design and optimiza-tion\cite{6,7,8,9} and so on. However, it has following inevitable disadvantages:

1) The poor computation efficiency. At the beginning, the algorithm converges quickly. However, while the solution is close to the optimal solution or subaltern optimal solution accepted, the convergence speed decreases quickly.

2) Prematurity in the algorithm. When the algorithm reaches local optimal solution, it can not jump out the area. Because binary-coded genetic algorithm can be easily implemented, it has been widely used in different areas. Nevertheless, binary-coded genetic algorithm usually does not meet the needs of actual projects:

3) Solution space discontinuity. Binary-coded can only express discrete values, which causes the decrease of computation precision.

4) The poor uniform standard. When computing fitness value, the binary value should be transformed to the real value firstly. Computation efficiency will decrease.

Based on these points, real-coded genetic algorithm appears, which can avoid some disadvantages of binary-coded genetic algorithm, but its inherent defects in algorithm also exist.

In order to improve the performance of GA, a lot of research has been conducted\cite{11,12,13}. Ortiz-Boyer et al.\cite{11} proposed a way to improve crossover operation of real-coded GA and was successfully applied to solve artificial problems. Chang\cite{12} contrived the multi-parents crossover to estimate the parameters of nonlinear process systems. Javadi et al.\cite{13} combined GA with neural network to deal with high dimension problems; meanwhile the computation efficiency was also improved.

In this paper, both mutation operation and crossover operation are improved to increase convergence speed and avoid prematurity. The organization of this paper is as follows. In section 2, the genetic algorithm steps are spread out. In section 3, we set forth the improvement on mutation operation and crossover operation. In section 4, two examples are given to testify the feasibility of IRGA. Conclusions are given in section 5.

2. Improved Genetic Algorithm Steps

To explain the computation process of the improved genetic algorithm, the authors analyze these steps as follows:

Step 1: Extract constraint variables and goal functions from the model and construct proper fitness function.

Step 2: Initialization. If the range of constraint variable is larger, we had better divide it into several small intervals. Because random initialization makes the number of population on the edge smaller, the optimal solution or subaltern optimal solution accepted may be lost. Theoretically, the larger the population size is, the less the chance of prematurity is. However, it will lead to the decrease of computation efficiency. Population size should be between several tens and several hundreds and define it as $t = 0$ generation.

Step 3: Selection operation. Calculate the fitness of each population unit and rank them by a fixed sequence. Select the first twenty elitists to prepare for the next mutation operation and crossover operation and make the first five elitists enter into the $t + 1$ generation directly.

Step 4: Mutation operation. Put the five relatively bad elitists into mutating pool and make them mutate according to a certain regulation so as to approach the relatively good elitists.

Manuscript received April 20, 2007; revised July 21, 2007.
Z.-L. Wang, P. Yang, D. Ling, and Q. Miao are with School of Mechatronics Engineering, University of Electronic Science and Technology of China, Chengdu, 610054, China, (e-mail: wzhlglai@uestc.edu.cn).
Step 5: Crossover operation. The population size increases rapidly and they are around the best elitists.

Step 6: Termination qualification. Given a precise $\delta$, if the fitness difference between $t$ and $t+1$ generation is less than or equal to $\delta$, that is

$$|F(X_1^{(0)}, X_2^{(0)}, \cdots, X_n^{(0)}) - F(X_1^{(0)}, X_2^{(0)}, \cdots, X_n^{(0)})| \leq \delta$$

where $X_1^{(0)}, X_2^{(0)}, \cdots, X_n^{(0)}$ and $X_1^{(t)}, X_2^{(t)}, \cdots, X_n^{(t)}$ represent the samples at generation $t$ and $t+1$ respectively; $F(\cdot)$ represents the fitness, then algorithm terminates, or else returns to step 3.

3. Improved Algorithm

In this section, improvements on algorithm are discussed. First, improved mutation operation is proposed based on gradient theory. Because in many practical fields, a lot of object functions are continuous and differentiable. We will realize how the improved mutation operation takes effect on the algorithm. Finally, we discuss how to improve the crossover operation to make the algorithm more efficient.

3.1 Mutation Operation Improvement

Let $X^{(t)} = [X_1^{(t)}, X_2^{(t)}, \cdots, X_i^{(t)}, \cdots, X_n^{(t)}]$, it is one of the five relatively bad elitists and $F(X^{(t)})$ is the fitness function, then

$$K_i = \frac{\partial F(X^{(t)})}{\partial X_i^{(t)}}$$

(1)

with $X_1^{(t)}, X_2^{(t)}, \cdots, X_i^{(t)}, \cdots, X_n^{(t)}$, $K_i$ can be reached.

$$\begin{align*}
X_i^{(t)}' &= X_i^{(t)} + \alpha, \quad K_i > 0 \\
X_i^{(t)}' &= X_i^{(t)} - \alpha, \quad K_i < 0
\end{align*}$$

(2)

where $\alpha = \frac{1}{2^r \times (K_i + 20)}$, $r$ is the evolutional generation.

So $X^{(t)}' = [X_1^{(t)}', X_2^{(t)}', \cdots, X_i^{(t)}', \cdots, X_n^{(t)}']$. Substitute $X^{(t)}$ for $X^{(t)}$, we have

$$K_i' = \frac{\partial F(X^{(t)'})}{\partial X_i^{(t)'}}.$$  

(3)

When $K_i \times K_i' > 0$, if $F(X^{(t)'}) < F(X^{(t)})$, $X_i(t)' = X_i(t)$; if $F(X^{(t)}) > F(X^{(t)'})$, (1) will be used.

When $K_i \times K_i' < 0$, if $F(X^{(t)'}) < F(X^{(t)})$, (1) will be used; if $F(X^{(t)}) > F(X^{(t)'})$, substitute $(1/2)\alpha$ for $\alpha$ until $K_i \times K_i' > 0$, then (2) will be used with the last $\alpha$.

3.2 Crossover Operation Improvement

According to literature review, SGA usually cannot converge to the global optimal solution. Only when relatively best elitists get into the next generation, does the convergence to the global optimal solution become possible[14]. Before crossover operation, the first five elitists become the individuals of the next generation.

Suppose that $X_m^{(t)}$ and $X_n^{(t)}$ are the two units in the $t$ generation. $d$ denotes the distance between $X_m^{(t)}$ and $X_n^{(t)}$; $\varepsilon$ denote the predetermined small number; $\beta$ is the parameter which can be given according to the requirement.

$$X_m^{(t)} = \left[ X_{m_1}^{(t)}, X_{m_2}^{(t)}, \cdots, X_{m_i}^{(t)}, \cdots, X_{m_n}^{(t)} \right]$$

$$X_n^{(t)} = \left[ X_{n_1}^{(t)}, X_{n_2}^{(t)}, \cdots, X_{n_i}^{(t)}, \cdots, X_{n_n}^{(t)} \right]$$

$$d = \left| X_m^{(t)} - X_n^{(t)} \right|.$$  

(4)

If $d \leq \varepsilon$ and $F(X_m^{(t)}) > F(X_n^{(t)})$, we can obtain

$$\begin{align*}
X_{m_i}^{(t)}' &= X_{m_i}^{(t)} + \frac{1}{\beta} \text{rand}(X_{m_i}^{(t)} - X_{n_i}^{(t)}) \\
X_{n_i}^{(t)}' &= X_{n_i}^{(t)} - \frac{1}{\beta} \text{rand}(X_{m_i}^{(t)} - X_{n_i}^{(t)}).
\end{align*}$$

(5)

Fig. 1. The flowchart of operations of Improved Real Genetic Algorithm.
If \( d \leq c \) and \( F(X_n^{(i)}) < F(X_m^{(i)}) \), we have

\[
\begin{align*}
X_{mi}^{(i)} &= X_{mi}^{(i)} + \frac{1}{\beta} \text{rand}(X_{mi}^{(i)} - X_{mi}^{(i)}) \\
X_{mi}^{(i)} &= X_{mi}^{(i)} - \frac{1}{\beta} \text{rand}(X_{mi}^{(i)} - X_{mi}^{(i)}).
\end{align*}
\]

(6)

One of filial generation lies between the two parents and the other lies beside the two parents but is close to the better one. Crossover operation not only has the characteristic of simple crossover operation, but also enlarges the searching space. However, it may cause some variables to go beyond the constraint range. Thus we make some rules as follows:

- If \( X_{mi}^{(i)} < a_i \), then \( X_{mi}^{(i)} = a_i \)
- If \( X_{mi}^{(i)} < a_i \), then \( X_{mi}^{(i)} = a_i \)
- If \( X_{mi}^{(i)} > b_i \), then \( X_{mi}^{(i)} = b_i \)
- If \( X_{mi}^{(i)} > b_i \), then \( X_{mi}^{(i)} = b_i \)

Because flowchart of selection operation and crossover operation are simple, we mainly describe the flowchart of mutation operation (when \( K' > 0 \)) in Fig. 1.

4 Case Study

4.1 Multi-Modal Function

Multi-modal function can be defined as

\[
\max f(x_1, x_2) = 21.5 + x_1 \sin(4\pi x_1) + x_2 \sin(20\pi x_2)
\]

s.t. \(-3.0 \leq x_1 \leq 12.1, \quad 4.1 \leq x_2 \leq 5.8\)

There are 510 apexes to function and it is easy to be trapped in local apexes.

We solve this problem with SRGA and IRGA, and the results are shown in Fig. 2 and Table 1.

Table 1: Comparison between SRGA and IRGA (Pop size =200)

<table>
<thead>
<tr>
<th>Generation</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>11.0704</td>
<td>11.0911</td>
<td>11.1143</td>
<td>11.1326</td>
</tr>
<tr>
<td>SRGA</td>
<td>5.7322</td>
<td>5.7251</td>
<td>5.7276</td>
<td>5.7262</td>
</tr>
<tr>
<td>( f(x_1,x_2) )</td>
<td>35.2295</td>
<td>37.3248</td>
<td>37.8653</td>
<td>38.0018</td>
</tr>
<tr>
<td></td>
<td>11.5739</td>
<td>11.5964</td>
<td>11.6014</td>
<td>11.6308</td>
</tr>
<tr>
<td>IRGA</td>
<td>5.7203</td>
<td>5.7221</td>
<td>5.7229</td>
<td>5.7286</td>
</tr>
<tr>
<td>( f(x_1,x_2) )</td>
<td>36.2412</td>
<td>37.9829</td>
<td>38.2681</td>
<td>38.6826</td>
</tr>
</tbody>
</table>

As shown in Fig. 2, with SRGA, the function reaches 38.3421, where \( x_1 = 11.1256 \) and \( x_2 = 5.7164 \) in generation 23. Meanwhile with IRGA, the function reaches 38.8453, where \( x_1 = 11.6255 \) and \( x_2 = 5.7248 \) in generation 7. Computation efficiency has increased by over 3 times and computation precision has been improved by 0.5132.

Fig. 2. Comparison between SRGA and IRGA.

4.2 A Case Study with Complicated Constraints

Given that a single-stage helical cylindrical gear reducer, parameters are described as power \( P = 22 \text{ kW} \), rotation speed \( n_1 = 970 \text{ r/m} \), speed ratio \( i = 4.6 \), single-way transmission and the life-span \( t = 10 \text{ years} \). The aim is to design the reducer and minimize the volume of it.

According to the condition given above, we choose 40Cr as the material of the driving gear and driven gear and make tooth width coefficient \( \psi = b / d = 0.8 \). Volume of the reducer can be written as

\[
V = \frac{0.8\pi m_1^3 x_1^3}{4 \cos^3 \beta} (1 + i^2) = 13.923 m_1^3 x_1^3 \cos^3 \beta.
\]

Let \( X = [x_1, x_2, x_3, \psi] = [m_1, x_1, \cos \beta] \), we have

\[
\min f(X) = 13.923 x_1^3 x_2^3 x_3^{-3}.
\]

Considering the undercut principle of the driving gear, the equation should satisfy

\[
g_1(X) = 17 - x_2 \leq 0.
\]

Helix angle should meet

\[
g_2(X) = x_3 - 0.9903 \leq 0, \quad g_3(X) = 0.9659 - x_3 \leq 0.
\]

Controlling gear module to transfer power:

\[
g_4(X) = 2 - x_1 \leq 0.
\]

Requiring on tooth width:

\[
g_5(X) = 16 - 0.8 x_2 x_3^{-1} \leq 0, \quad g_6(X) = 0.8 x_2 x_3^{-1} - 35 \leq 0.
\]

Touch fatigue intensity should be satisfied

\[
404132 x_1^3 x_2^3 x_3^{-1} = 1170 \leq 0.
\]

Bend fatigue intensity should satisfy

\[
2810702.8 x_1^{-3} x_2^{-2} x_3^2 - 528.6 \leq 0, \quad 2635413 x_1^{-3} x_2^{-2} x_3^2 - 514.3 \leq 0.
\]

Such kind of function is commonly used in many fields.
The function needs many evolutational generations to reach the optimal solution based on SRGA, but with IRGA the number of generation reduce and it is easy to reach global optimal solution.

Table 2: Show the difference between SRGA and IRGA

<table>
<thead>
<tr>
<th></th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>f(x)</th>
<th>Generation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRGA</td>
<td>2.386</td>
<td>19.933</td>
<td>0.966</td>
<td>1.6618e+006</td>
<td>40</td>
</tr>
<tr>
<td>IRGA</td>
<td>2.3061</td>
<td>21.0569</td>
<td>0.9864</td>
<td>1.6611e+006</td>
<td>6</td>
</tr>
</tbody>
</table>

5. Conclusions

IRGA has the following advantages compared with SRGA. 1) When the initializing population size, the constraint range is divided into small intervals and initialized respectively, IRGA will reduce the probability of being trapped in the local optimal solution. 2) IRGA ensure the relatively bad individuals to be close to the better by improving mutation operation and so increase the algorithm efficiency. 3) IRGA not only has the characteristic of simple crossover operation, but also extends the searching space by improving crossover operation. 4) When dealing with multi-modal functions, IRGA demonstrates its superiority.

However, IRGA still has some problems that need further improvement. Some parameters cannot be given according to a certain uniform criterion, such as $\varepsilon$ and $\beta$.

When dealing with functions with complicated constraints, some variables need to be further adjusted on the basis of the constraint range.

References


Zhong-Lai Wang was born in Shandong Province, China, 1980. He received his B.S. degree from Sichuan University in 2004. He is currently a Ph.D. candidate with School of Mechatronics Engineering, University of Electronic Science and Technology of China (UESTC). His research interests include reliability engineering, optimization design, life cycle reliability- based design and robust design.

Ping Yang was born in Sichuan Province, China, 1963. He is currently a professor with School of Mechatronics Engineering, UESTC. His research interests include optimization design and fault diagnosis.

Dan Ling was born in Liaoing Province, China, 1975. She is currently an associate professor with School of Mechatronics Engineering, UESTC. Her research interests include reliability engineering maintenance.

Qiang Miao was born in Hubei Province, China, 1976. He is currently an associate professor with School of Mechatronics Engineering, UESTC. His research interests include fault diagnosis and maintenance.