Automatic Test Data Generation for Path Testing Using Genetic Algorithms

Wang Xibo, Su Na

Shenyang University of Technology, Shenyang, Liaoning, 110870, China

Abstract—**Software testing is the important means that guarantee software quality and reliability. Improving the automation ability of software testing is very important for ensuring software's quality and reducing development cost, and improving the automation ability of test cases generation is the key point for the entire process. This paper discusses the methods and techniques of genetic algorithm as the key algorithm to automatically generating the test data, and elaborates some specific problems need to solve in realization process: such as coding, the selection of fitness function and the improvement of hereditary operator, etc.**

Keywords-Genetic Algorithm; Path Testing; Automatic Test Data; Branch-Cover

I. INTRODUCTION

The automatic test data generation plays crucial roles in the process of software testing in the automation. Path testing is the major test way of the software structure testing, and the key point to resolve the path testing is to find specific input data and make it possible to cover a path in the testes program. Therefore, how to automatically generate the required test data has become the main research question. At present, in automatic test data generation field, the more common methods are such as： notation to perform, random method, iteration relaxations and heuristic. But their common deficiencies in the procedures are difficult to generate test data of the big and complicated programs.

Genetic algorithm as an effective global and smart search method, it reveals its own strength and efficiency to solve the larger space, nonlinear, optimized for high complicated problems, and thus it gradually been introduced into the test field ,which provides a new method to solve the problems of generated test data. In the path testing of test data generation, now widely used method is based on genetic algorithm or genetic algorithms to combine with other algorithm to the new algorithm. In the literature [1], it uses the genetic algorithm to the path covered in software testing of test data automatically generation the of Ada structure, and got the conclusion that genetic algorithm is more efficient than climbing method and random method in generating test data. In the literature [2],the author considers the characteristics of test data automatically generation to combine the genetic algorithms and simulated annealing algorithms, making full use of the advantages of their respective global optimization capability and local optimization capability to improve the generation capacity of test data.

Based on the present generation technology of test case and basic principle of genetic algorithm, this paper designs the test data generation system based on path testing, focusing on the generation efficiency of test data. In addition, the author also makes some improvement to the fitness function and genetic operations to enhance the usefulness of the algorithm.

II. THE TEST DATA SYSTEM DESIGN OF PATH TESTING BASED ON GENETIC ALGORITHM

A.. Gentic Algorithm

Genetic algorithm is a computational model simulating the biological evolution process of the genetic selection theory of Darwin. It was first came up by the professor j. Holland in 1975 in the university of Michigan. Its main characteristic is the direct operation to the structure object, there is no qualified of the report and functions, continuity of the parallelism and global optimization capability, automatically acquiring and guiding optimization space by cost-effective way, readjust optimization direction the way and not need to fixed rules. These natures of genetic algorithms are widely used in biology, engineering, metric, computer science, image processing, pattern recognition and other areas.

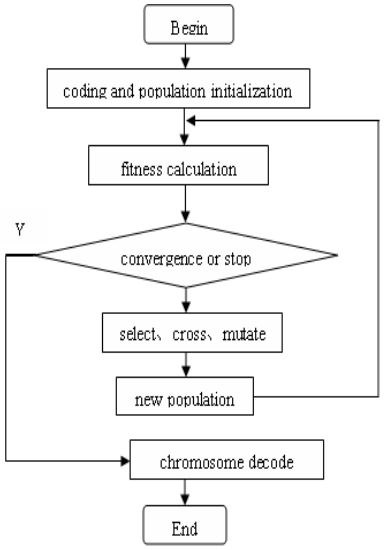

Figure 1. The flow chart of genetic algorithm

Genetic algorithm is a "generation and inspection" iterative process optimization algorithm that is a group of operation and all the individuals in the group are the operation object. Selection, cross and mutation are the

978-0-7695-4296-6/11 $26.00 © 2011 IEEE
DOI 10.1109/ICMTMA.2011.152

major operation of the genetic algorithms, which constitute genetic operation containing all properties other conventional algorithms not exist. There are five basic operation of genetic algorithm in the following：

- parameter coding
- the design of fitness function
- the design of genetic operation
- The set of controlling parameters (refers the size of population, number of iterations, etc.)

B. The design process of the system

The system model of automatic test data generation based on genetic algorithms facing path testing is shown in fig 2 model. The main work is automatically generating test data to cover the branch in the tested program.

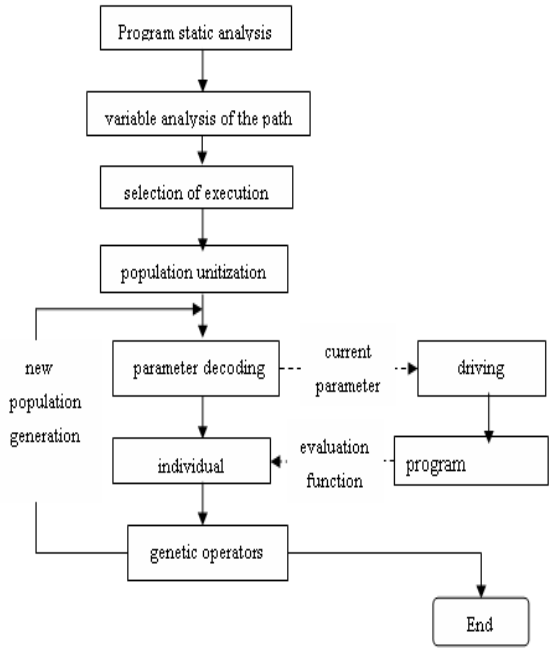

Figure 2. The system model of automatic test data generation

- According to white-box testing, we can make a static analysis in the tested program, analyze the variables affecting the program execution path, choose the test path and at last determine the initial population and coding style.
- Randomly generate the first generation of population, then map the bit string of the individual into the parameter values and transfer the tested program driving it to be executed.
- Using the technology of program instrumented, insert the fitness function assessed the parameters in tested program and return the function value to genetic algorithm.
- Analyze the fitness function value, using the three major operation of selection, cross and mutation change the structure of individual string and generate new group.
- Map the new generated individual string into a new test data, repeat steps 3), until meet the requirements of the fitness function value or achieve specified number of iterations and can cover the designated path branch which the user need test.

Figure 3. The static analysis of the tested program

III. THE ALGORITHM DESIGN OF SYSTEM

A. Parameter Coding

Under the principle of "only useful parameter for coding", we only code the variables which are relevant to the expression in the path and the other variables are not be encoded. Because there are multiple parameters in the program, so use "many parameters cascading code", for each parameter is to be coded into binary code string and then cascade all the bit strings of parameters get a many parameters of the code string, which is shown in the following:

Cascading before

$$X_1 \quad X_2 \quad \ldots\ldots \quad X_n$$

$$a_{11}a_{12}\ldots a_{1m}\ a_{21}a_{22}\ldots a_{2m}\ \ldots\ldots\ a_{n1}a_{n2}\ldots a_{nm}$$

Cascading behind

$$a_{11}a_{12}\ldots a_{1m}\ a_{21}a_{22}\ldots a_{2m}\ \ldots\ldots\ a_{n1}a_{n2}\ldots a_{nm}$$

Including $a_{ij} \in \{0, 1\}$, $i \in [1, n]$, $j \in [1, n]$

The cascading string is a unique individual in population of the genetic algorithms. When decodes, cut to n yards length of m from whole code length, and then separately decodes.

B. The structure of fitness function

The fitness function in algorithms directly affects convergent speed of the genetic algorithms and whether can find the best solution. Good fitness function can improve the search efficiency of best solution and achieve the purpose.

At present, the most frequently used method for designing fitness function is adding all the branch functions of the chosen path at one time. For example: if a program executes one path through the branches F_1, F_2 to F_n, then the fitness function of the path is：

$$F=F_1+F_2+\ldots+F_n$$

Although the predicate function plays an important role in generating test data, but there are still inadequate. This paper makes some improvement to this method, first take a different scaling factor for every branch, and put them together until they find the same order of magnitude of the comparison, this can better reflect the parameters of degree. Therefore, the fitness function equation becomes:

$$F=a_1F_1+a_2F_2+\ldots+a_n F_n$$

If a branch function is not executed in the program's dynamic execution, which the fitness function as the branch of the path has not been implemented, so the value of scaling factor will become zero. Scaling factor makes the fitness function more reasonable. Thus, whether the parameter accurate the tested path to meet the condition in the form of quantification and through fitness functions determine the performance of the parameter.

Now the following is a particular example that the combination of branch function structure and process of program instrument.

```
   void Triangle (int x ; int y ; int z)
{  if (a > 1 & & a < 1000)
       if (b > 1 & & b < 1000)
          if (c > 1 & & c< 1000)
             if ((a= =b)&&(b= =c))
   printf ( "This is a equilateral triangle." ) ;
}
```

It is easy to see that there are five paths of the sample program. The present objective is to find that all the test data which can cover the all real branches in the tested program. And the program which after been instrumented is in the below:

```
     float Right Triangle (int x ,int y ,int z)
{    float f1 ,f2 ,f3 ,f4 ,F ;
     int a1,a2,a3;
     f1 = max (1 – a, a - 1000) ;
    / * the instrument of branch function*/
       if (a > 1 & & a < 1000)
    f2 = max (1- b, b - 1000) ;
    / * the instrument of branch function* /
       if (b > 1 & & b < 1000)
     f3 = max (1 - c, c- 1000);
    / * the instrument of branch function* /
       if (c> 1 & & c < 1000)
     f4 = max (abs (a-b), abs (b-c));
   / * the instrument of branch function* /
       if ((a= =b)&&(b= =c))
      printf ( "This is a equilateral triangle." ) ;
      if (f1 < 0) f1 = 0 ;
      if (f2 < 0) f2 = 0 ;
      if (f3 < 0) f3 = 0 ;
     F = a1f1 + a2f2 + a3f3 + a4f4 ;
  / * the instrument of fitness function* /
      return F ;
  / * return to the packet of genetic algorithm* /
}
```

C. The improvement of genetic operation

1) Selection operator

By selecting the individuals in the population which with large fitness will have greater opportunities exist than the small ones. In the system of path testing, retain the best individuals in the current and replace the bad individuals with best individuals.

2) Crossover operator

This paper makes some improvement in crossing operation on the basic of uniform crossing. When the individuals have many crossing point, the benefit of individual was to undermine possibilities for comparison, so the establishment of a control value of c, and randomly generated shielding word binary representation of a number of the percentage of the total length equal to the control values.

3) Mutation operator

If the two individuals have the same result before and after crossing operation, so make this individual mutate and generate new individuals. This can expand the searching space and avoid causing local convergence.

Because the genetic algorithm has much more advantage than of traditional algorithm, which makes it have particular advantage in solution path testing.

Through some improvement of genetic algorithm, the search capability and path coverage have been enhanced a lot.

IV. THE UNIT-TESTING FRAMEWORK OF CPPUNIT

The test case in software testing including two parts: the input data and expected results.But the automatically generated program of test case only can generate the input data.Therefore, the expected results need to manually calculate or use the test specification to get.

Cppunit is a functional test framework specially facing unit testing, finishing black box testing and focusing on the test results. By way of writing test driver program, the cppunit framework will take test data as the input data and finish the program testing in the framework. Comparing the execution results with the expected results, it can judge whether the test case has passed the test. The defect of the cppunit framework is that test data must be manual input, which will increase the number of test and can not guarantee every path in the tested program will be tested. It only gets something that whether the test results meet the needs of test specification. In the paper, the system achieve that automatically generate test data, focusing on the relative paths input data and the data itself. Therefore, it uses the cppunit framework to finish the functional test in the paper, validating the test data whether meet the expected results.

V. CONCLUSIONS

In software testing, the generation of testing data is one of the key steps which has a great effect on the automation of software testing. In order to increase the efficiency of testing data generation, this paper presents a newly improved genetic algorithm for generating software test

data. This paper makes some improvement in traditional genetic algorithm and adopts fitness scaling algorithm. Results of tests show that the algorithm can avoid precocious phenomena to a certain extent and it has higher rate of convergence. And the method has some practical values in path testing and is valuable to be investigated further.

REFERENCES

[1] HaoWang, JunkaiXie, hongyiGao. The application of genetic algorithms in software testing data generation. Computer engineering and applications.2004(in Chinese

[2] BoFu. Testing data automatically generation based on SGA. Computer engineering and applications.2005(in Chinese)

[3] MyerG, SandlerC, BadgettT, etal. The Art of Software Testing[M]:Second Edition. John Wiley & Sons, June21, 2004.

[4] LifuWang. Software Engineering[M].2002

[5] YajuanSun. Test data automatically generation based on genetic algorithm[D].Hebei university of technology,2006.(in Chinese)

[6] XiaopingWang, LimingCao. The genetic algorithm[m].Sian jiaotong University Press,2002.

[7] Zhang J, Xu C, Wang X. Path - oriented Test Data Generation Using Symbolic Execution and Constraint Solving Techniques [C]. Beijing: Proc. of the Int1Conf .on Soft ware Engineering and For mal Methods, IEEE Computer Society Press, 2004: 242 - 250