Cloud computing environment based on Web log mining algorithm implementation of test

Wang Bin  
College of Information Science & Electronic Technology  
Jiamusi University  
Jiamusi, China  
e-mail:jmsuwang@163.com

Yong-cheng Jiang* (Author for correspondence)  
College of Mechanical Engineering, Jiamusi University  
Jiamusi, 154007, China;  
E-mail: jiangyongcheng@126.com

Li jing  
College of Information Science & Electronic Technology  
Jiamusi University  
Jiamusi, China  
e-mail: lijing2483@163.com

Abstract—Software test is the important means that guarantee software quality and reliability, and in this respect, it plays the role that other method cannot replace. However software test is a complex process, it needs to consume huge manpower, material resources and time, which takes the 40%-50% of entire software development cost approximately. Paper presents the inherent in software test case designing based on genetic algorithm is using genetic algorithm to solve a set of optimization test cases, and the framework includes three parts which are test environment construction, genetic algorithm and the environment for test.

Keywords- software test , test case , genetic algorithm

I. INTRODUCTION

Algorithm (genetic algorithm) is the simulation of Darwin's selection and the natural elimination process of biological evolution model. Web log mining algorithm was first proposed by Professor J.Holland in 1975 by the United States. Its main characteristic is the direct operation to structural object, there is no derivation and the continuity of function limit; inherent implicit parallelism and better ability of global searching; using the probability optimization method, can automatically access and guide optimal search space, adaptively adjust the search direction, does not need to determine the rules. The algorithm processes include coding, generation of initial population, fitness value was calculated, selection, crossover, mutation operation. The general flow diagram as shown in figure 1:

II. BASED ON THE WEB LOG MINING ALGORITHM FOR GENERATING TEST CASE

Based on the algorithm of test case generation basic processes as follows:
(1) analysis of the source code, access to program control flow diagram;
(2) by a program control flow diagram to get program branching path set, choose the target path;
(3) according to each predicate terms to the program instrumentation and the development of fitness function;
(4) set the algorithm parameters, including population size, variation, randomly generated initial test data set;
(5) using the test data to perform after plug-in source code, to obtain the fitness value, according to the fitness value judgment, if meet the program termination condition output results and exit the program, if the conditions are not satisfied to step (6);
(6) according to the fitness value, using the algorithm selection, crossover, mutation operation, generating new test data, and go back to step (5), repeat;

Program frame as shown in Figure 2, including the test environment structure, algorithm and test running environment part three.

Figure 2. the program frame diagram

A. The program to be tested into

The triangle classification procedure as an example to verify the algorithm. The triangle classification procedure in many software testing research is as the benchmark program to use, because it contains clear and complex logic, and even a larger range of integers as input, and only a small amount of input combinations to meet specific branch code. For example, when the input is 1 to 10 integer, 1000 input 10 groups only can satisfy the judgement as " equilateral triangle " branch. Triangular source for:

```java
String tri_type(int a,int b,int c) {
   string type;
```
if(a>b) change(a,b);
if(a>c) change(a,c);
if(b>c) change(b,c);
if(a+b>c)
{
    if(a==b||b==c)
    {
        if(a==c) type="Equilateral triangle";
        else type="An isosceles triangle";
    }
    else type="Ordinary triangle";
} else type="Is not a triangle";
return type; }

B. program flow analysis

The program flow diagram as shown in Figure 3, the program is divided into two parts, the first input value from the sort, then classify triangles.

![Figure 3. the triangle classification process](image)

C. path analysis

Through the process of analysis, the procedure to classify triangles statement as in Figure 3 is marked with a "1" to "6" on the branch, the branch path analysis as shown in table I:

<table>
<thead>
<tr>
<th>Path</th>
<th>Branch</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1</td>
<td>②</td>
<td>Is not a triangle</td>
</tr>
<tr>
<td>W2</td>
<td>①-④</td>
<td>Ordinary triangle</td>
</tr>
<tr>
<td>W3</td>
<td>①-③-⑤</td>
<td>An isosceles triangle</td>
</tr>
<tr>
<td>W4</td>
<td>①-③-⑥</td>
<td>Equilateral triangle</td>
</tr>
</tbody>
</table>

IV. CODE INSTRUMENTATION

String tri_type(int a,int b,int c)

{ string type;
if(a>b) change(a,b);
if(a>c) change(a,c);
if(b>c) change(b,c);
f1=c-(a+b) ;                      // Pile plug 1
if(a+b>c)
{
    f2=min(abs(a-b),abs(b-c));        // Pile plug 2
    if(a==b||b==c)
    {
        f3=abs(a-c);                    // Pile plug 3
        if(a==c) type="Equilateral triangle";
        else type="An isosceles triangle";
    }
    else type="Ordinary triangle";
} else type="Is not a triangle";
return type; }

V. RESULTS ANALYSIS

The triangle classification procedures are condemned as an equilateral triangle " path W4 for example, run the program, found in the 51 generation to find the optimal solution, table 2, 3 respectively for the zeroth generation, thirteenth generation, the fifty-first generation of fitness for the top 10 individuals case:

From the two table is visible, along with the evolution process continued, the overall fitness in the description of the algorithm increases, are moving towards the optimal solution in the direction of convergence, to find the optimal solution. This algorithm in the program of automatic test data generation is has a role, it will gradually improve individual.

ACKNOWLEDGMENT

This research was supported by the National Nature Foundation of China (Grant No. 61002004), the National Science Foundation of Heilongjiang Province (Grant No. E200908), the Natural Science Foundation of Heilongjiang Province Education Office(Grant No. 12511551), the National Science Foundation of Jiamusi University(Grant No. L2012-074), the National Science Foundation of Jiamusi City(Grant No. 12004) all support is gratefully acknowledged.

REFERENCES

### TABLE II. THE ZEROTH GENERATION TO THE TOP 10 INDIVIDUALS

<table>
<thead>
<tr>
<th>Individual number</th>
<th>Chromosome encoding</th>
<th>Parameter value</th>
<th>Fitness</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>1</td>
<td>110101011100111000</td>
<td>53</td>
<td>28</td>
</tr>
<tr>
<td>2</td>
<td>100110101001100111</td>
<td>38</td>
<td>41</td>
</tr>
<tr>
<td>3</td>
<td>01101010100101010</td>
<td>27</td>
<td>26</td>
</tr>
<tr>
<td>4</td>
<td>000111001000110000</td>
<td>7</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>100110101101101100</td>
<td>38</td>
<td>45</td>
</tr>
<tr>
<td>6</td>
<td>011011100111100010</td>
<td>27</td>
<td>35</td>
</tr>
<tr>
<td>7</td>
<td>111011111110101011</td>
<td>59</td>
<td>58</td>
</tr>
<tr>
<td>8</td>
<td>110111111000111011</td>
<td>54</td>
<td>56</td>
</tr>
<tr>
<td>9</td>
<td>100001101111001100</td>
<td>33</td>
<td>30</td>
</tr>
<tr>
<td>10</td>
<td>110011111000111010</td>
<td>51</td>
<td>56</td>
</tr>
</tbody>
</table>

### TABLE III. THE THIRTEENTH GENERATION TO THE TOP 10 INDIVIDUALS

<table>
<thead>
<tr>
<th>Individual number</th>
<th>Chromosome encoding</th>
<th>Parameter value</th>
<th>Fitness</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>1</td>
<td>010000001011010000</td>
<td>16</td>
<td>11</td>
</tr>
<tr>
<td>2</td>
<td>101000111101101000</td>
<td>40</td>
<td>61</td>
</tr>
<tr>
<td>3</td>
<td>101000101000110010</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>4</td>
<td>101011001011101111</td>
<td>43</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>010100010110111100</td>
<td>20</td>
<td>23</td>
</tr>
<tr>
<td>6</td>
<td>111011011010111111</td>
<td>61</td>
<td>44</td>
</tr>
<tr>
<td>7</td>
<td>100111110110001010</td>
<td>39</td>
<td>38</td>
</tr>
<tr>
<td>8</td>
<td>011101101111101000</td>
<td>26</td>
<td>61</td>
</tr>
<tr>
<td>9</td>
<td>000011110101101000</td>
<td>3</td>
<td>41</td>
</tr>
<tr>
<td>10</td>
<td>011111100101100111</td>
<td>31</td>
<td>37</td>
</tr>
</tbody>
</table>
Start

Gen:=0

Determine the length the string L

A randomly generated initial population M

j:=0

Choose individual mutation

Implementation variation

The variability of results add to a new group

j:=j+1

j=PmM?

Y

N

Output results

results

Fitness

Genetic operator

j:=0

Select two Exchange individual

Implementation of exchange

Two new individuals are added to the new group

j:=j+1

j=PcM?

N

Y

j:=0

Copy

Added to the new group

j:=j+1

j=P,M?

N

Y

j:=Gen+1

Choice Replication of individual

Gen:=Gen+1

Ps

Pc

Pm

Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013)

Published by Atlantis Press, Paris, France.
© the authors

3168

Figure 1. algorithm