Automatic Test Data Generation Based On Fuzzy Logic

Amir S. Almallah and Ismael Abdulsattar
Dep. of computer science.Collage of science, University of Mustansiriya
Received 26/3/2013 – Accepted 15/9/2013

ABSTRACT

The complexity of software systems has been increasing dramatically in the past decade, and software testing as a labor-intensive component is becoming more and more expensive. With the complexity of the software, the cost of testing software is also increased. Thus with automatic test data generation the cost of testing will dramatically be reduced. This paper uses fuzzy logic concepts to generate test data automatically and this data will be used for the future to feed software which used in fuzzy logic applications like(Industrial automation, decision making process, such as signal processing or data analysis...etc). Software testing is probably the most complex task in the software development cycle. It is one of the most time-consuming, costing and frustrating process.

1. INTRODUCTION

Software testing is a process, which is used to identify the correctness, completeness and quality of a software. [1]. The effective generation of test data is one of the most difficult and expensive problems in software testing. Test data generation is the process of creating program inputs that satisfy some testing criterion [2]. Obviously, manually developing a large test data set to satisfy a testing criterion is usually expensive, laborious, difficult and error-prone. If test data could be automatically generated, the cost of software testing would be significantly reduced. It is usually observed that the input data near the boundary of a domain are more sensitive to program faults and should be carefully checked. A domain testing strategy is very effective in verifying the correctness of the boundary of a path domain; however, such a domain strategy is hard to implement since the strategy requires test data generated on and near the boundary, and the test generation is more difficult when some of the constraints are nonlinear or in a discrete space [3]. In general automatic test data generator are contain big challenges especially in the huge software that could make any decision based on fuzzy data here the challenges will be more[4][5]. Test Data Generation (TDG) is crucial for software testing because test data is one of the key factors for determining the quality of any software test during its execution [7]
In the fuzzy logic concept if there is no interference among the generated data intervals that means no fuzziness to solve and it would be useless data generated to participate as input for fuzzy applications programs.[6]

But the huge mathematics problem will rise in this case which is "what are the thresholds values to stop generating" as we know all the close intervals contains two boundaries start number and end number. let have close intervals a, b
Symbols: \( a \leq x \leq b \) or \([a, b]\)
how many numbers between \( a \) and \( b \) need to be generated to find the intersection with other closed intervals to decide this two intervals useful to feed fuzzy program or useless then we could eliminate it from the test data generator. This mechanism will play good threshold condition for filtering our generated data, for instance if we have the following two intervals \( \alpha \) and \( \beta \) as

In the above figure we see that the whole interval \( \beta[c, d] \) intersect with \( \alpha[a, b] \) in such case the generated interval will be useful but in the following case will show how it is useless input that we should eliminate it

In the above cases we show how the intersection will work in the close intervals but the intersection is only one of basic concept in fuzzy logic due to the fuzzy set has intersection with each other with degree of membership for each element in the fuzzy set.

2. Fuzzy logic and fuzzy set

In figure (1) showing below we have the following sets A \([-4, 4]\) and B \([0, 8]\) and they are intersect in the interval \([0, 4]\) these two intervals will be useful as input for fuzzy program
Fuzzy set can have different mechanisms to implement but they are all have the same concept, on the other hand if there is no intersection it will be useless pair to consider as input for fuzzy program.

### 3. Proposed Mechanism

Here we create three factors $\alpha, \epsilon, \beta$ for filtering test data generated form random data generator that generate random intervals the rule of the first factor $\alpha$ will decide which two or more interval useful or not by found if there is an intersection among them or not, the rule of the second factor $\epsilon$ is to determine the value of intersection that will decide based on user demand or program requirements, and the third one is to decide the maximum number (capacity) that the intervals could take which also can be determined based on user demand that can fulfill user requirement. As show in the figure (2):

![Figure 1: Two fuzzy set A and B with degree of membership](image-url)
4. Experimental Result

As the random generator generate random pairs of intervals we run the program several times with different possible values of our factors $\alpha, \varepsilon$ and $\beta$ and show the effect of that factors over generated intervals. We can see clearly here if the value of $\alpha$ equal to zero that is mean there is no intersection among the generated interval, and we can take crisp decision that will be useless interval pair and if it is one that means there is an intersection values then move to the next factor $\varepsilon$ to test how much the minimum ratio required to take in our account and third factor is $\beta$ use to determine how much possible degree you divide the intersection intervals. Thus we can see the big effect of that three factors to filterize random generated intervals and eliminate the useless interval pairs as shown in the following table.
Table-1:

<table>
<thead>
<tr>
<th># generated intervals</th>
<th>$\alpha$ value</th>
<th>$\epsilon$ value</th>
<th>$\beta$ value</th>
<th># useful intervals</th>
</tr>
</thead>
<tbody>
<tr>
<td>10582</td>
<td>1</td>
<td>0.2</td>
<td>27</td>
<td>7010</td>
</tr>
<tr>
<td>10110</td>
<td>1</td>
<td>0.15</td>
<td>52</td>
<td>7301</td>
</tr>
<tr>
<td>12001</td>
<td>0</td>
<td>0.4</td>
<td>42</td>
<td>0</td>
</tr>
<tr>
<td>15040</td>
<td>1</td>
<td>0.5</td>
<td>20</td>
<td>1560</td>
</tr>
<tr>
<td>17080</td>
<td>1</td>
<td>0.13</td>
<td>48</td>
<td>8120</td>
</tr>
<tr>
<td>18090</td>
<td>1</td>
<td>0.029</td>
<td>41</td>
<td>8601</td>
</tr>
<tr>
<td>20100</td>
<td>1</td>
<td>0.02</td>
<td>49</td>
<td>9092</td>
</tr>
<tr>
<td>21988</td>
<td>1</td>
<td>0.015</td>
<td>50</td>
<td>10906</td>
</tr>
<tr>
<td>22078</td>
<td>0</td>
<td>0.0011</td>
<td>0.0023</td>
<td>0</td>
</tr>
</tbody>
</table>

Table-1: and figure(3) showing the affection of $\alpha$, $\epsilon$, and $\beta$ factors on the generated intervals. We can see the big filtering process done over the random generated intervals.

Figure-3: shows the automatic test data generator and useful one of it

5. Suggestions and Conclusion

The automatic test data generation is huge concept to deal with and our invent mechanism to use not normal data but fuzzy data that would use to test fuzzy software in all its applications and we can see how three filters did significant reduction over the automatic random generated data. This approach also can fit the crisp decision software by manipulating the second and third factor. We suggest as future work to take adaptive factor that will select based on program that we want to test it that will be great factor can we use to determine the useful path will take to have good test data generator.
REFERENCES