An Extensive Approach on Random Testing with Add-On Technique for Novel Test Case

M. Jaikumar and Ramya Mohan

Abstract--- This paper proposes the use of Centroidal Voronoi Tessellations (CVT) to address the existing problem of Adaptive Random Testing and Quasi Random Testing in order to maximize the test case coverage. Accordingly a Novel Random Border CVT (NRBCVT), is proposed which can enhance the previous Random Border Centroidal Voronoi Tessellation by performing k-means Clustering to generate initial cluster centers instead of selecting random points as centroids. This leads to selection of more test cases than the RBCVT methods to improve their coverage of the input space. The generated test cases by the other methods act as the input to the NRBCVT algorithm and the output is an improved set of test cases. Therefore, NRBCVT is not an independent method and is considered as an add-on to the previous methods. Results from the experimental frameworks demonstrate that NRBCVT outperforms previous methods.

Index Terms--- Centroidal Voronoi Tessellations, K-means, Adaptive Random Testing, Quasi Random Testing, Test Cases

I. INTRODUCTION

Test cases are developed using various test techniques to achieve more effective testing. By this, software completeness is provided and conditions of testing which get the greatest probability of finding errors are chosen [1]. So, testers do not guess which test cases to chose, and test techniques enable them to design testing conditions in a systematic way [2]. Also, if one combines all sorts of existing test techniques, one will obtain better results rather if one uses just one test technique [6-12].

The main motivation of this proposed work is to generate high number of test cases than the traditional random testing techniques. The proposed method is easy to implement and has a low computational cost. Since an RT strategy uses random test cases, the input test cases may not cover all the regions of the input domain resulting in poor failure detection. In this research, it proposes a new test case generation approach, namely, Novel Random Border Centroidal Voronoi Tessellations (NRBCVT) using preclustering using k-means clustering instead of starting with random cluster, which utilizes Centroidal Voronoi Tessellations (CVT). The proposed NRBCVT approach enhances the existing state-of-the-art test case generation techniques. The performance of proposed work is based on the different failure rates.

II. RELATED WORK

T. Chen et al [3] presented in their paper a new ART method using the notion of iterative partitioning. The input domain is divided into equally sized cells by a grid. The grid cells are categorized into three different groups according to their relative locations to successful test cases. I. Ciupa et al [4] have developed a notion of distance between objects and a new testing strategy called ARTOO, which selects as inputs objects that have the highest average distance to those already used as test inputs. ARTOO has been implemented as part of a tool for automated testing of object-oriented software. They present the ARTOO concepts, their implementation, and a set of experimental results of its application. Analysis of the results shows in particular that, compared to a directed random strategy, ARTOO reduces the number of tests generated until the first fault is found, in some cases by as much as two orders of magnitude. ARTOO also uncovers faults that the random strategy does not find in the time allotted, and its performance is more predictable. In the paper H. Chi and E.L. Jones[5] presents work on generation of specification-driven test cases based on quasi random (low-discrepancy) sequences instead of pseudorandom numbers. This approach is novel in software testing. This enhanced uniformity of quasi random sequences leads to faster generation of test cases covering all possibilities. Monte Carlo [6] simulations are stochastic simulations and are always based on a Random Number Generator (RNG). Running the RNG constitutes often a non negligible part of the simulation time. The “Unrolling” technique is a way to optimize the access to random numbers. This method consists in a pre-generation and storage of random numbers in an array of values. This array is directly included into the binary code during the compilation process [17].

III. PROPOSED TEST CASE GENERATION APPROACH

3.1 Random Border CVT

The proposed novel RBCVT test case generation approach, which removes the undesirable feature of the CVT[7] In this regard, we propose a RBCVT calculation approach and investigate its associated runtime order. In addition, we propose a novel search algorithm to reduce the computational complexity of RBCVT. RBCVT is based on defining an imaginary random border outside the real borders of I. In this regard, we introduce a set of random points (R) in H which simulate an imaginary random border, as discussed in the next section. In Fig. 4, a set of RBCVT test cases is
demonstrated as well as the random border points in H. As indicated in this figure, RBCVT effectively removes the aforementioned undesirable feature of the CVT. Accordingly, Figure 1 indicates the generator points of RBCVT in the left-hand side and the resultant RBCVT points on the right-hand side.

![ART, RBCVT](image)

![QRT, RBCVT](image)

Fig. 1: Displays Points Generation using ART, QRT and RBCVT

### 3.2 RBCVT Calculation Method

To calculate the RBCVT test cases using a set of generator points, we propose a probabilistic method as follows:

1. **Step 1:** Determine the initial set of \( T = \{ t_i \}_{i=1}^{|T|} \) as generators, \( t_i \in I \), where \( i = 1; \ldots; |T| \).

2. **Step 2:** Initialize a random border point set of \( R = \{ r_n \}_{n=1}^{|R|} \), \( n=1 \) in which \( r_n \in H \), where \( n = 1; \ldots; |R| \). In addition, the combination of \( T \) and \( R \) is defined as \( TR = T \cup R = \{ trm \}_{m=1}^{|TR|} \).

3. **Step 3:** Random Point Selection Using K-Means Clustering

   Given a set of observations \((x_1, x_2, \ldots, x_n)\), where each observation is a \( d \)-dimensional real vector, \( k \)-means clustering aims to partition the \( n \) observations into \( k \) sets \((k \leq n)\) \( S = \{ S_1, S_2, \ldots, S_k \} \) so as to minimize the within-cluster sum of squares (WCSS):

   \[
   s \sum_{i=1}^{|R|} \sum_{x_j \in S_i} \| x_j - \mu_i \|^2
   \]

   where \( \mu_i \) is the mean of points in \( S_i \).

4. **Step 4:** Initialize a random background point set of \( B = \{ b_j \}_{j=1}^{|B|} \), \( j=1 \) in which \( b_j \in (I \cup H) \), where \( j = 1; \ldots; |B| \).

5. **Step 5:** Cluster the \( B \) into \(|TR| \) cells such that \( b_j \in Vm \), \( trm = \beta(b_j, TR) \)

6. **Step 6:** Calculate the centroids of Voronoi regions only for those \( Vm \) where the generator belongs to \( T \), denoted by \( V_i \) (we do not need to update border points). For the probabilistic approach, (8) is simplified to

   \[
   t_i^* = \frac{\sum_{b_j \in F_i} b_j}{\sum_{b_j \in F_i} 1}
   \]

   where \( \rho \) is set to a unit value in this application.

7. **Step 7:** Update the generators, \( t_i \), where \( i = 1; \ldots; |T| \) are replaced with the corresponding \( t_i^* \).

8. **Step 8:** Go to step 3 until the stoppage criterion is met

   A stopping criterion can be

1) The distortion value between \( t_i \) and \( t_i^* \), \( i = 1; 2; \ldots, |T| \), in each iteration, is reduced to less than a threshold, or

2) A constant number of iterations. Within this study, a constant number of 10 iterations have been selected. This stopping criterion was selected due to its perceived convergence among all trial runs of the algorithm.

### 3.3 Preprocessing Step

This section explains step 3 of the RBCVT-Fast algorithm that is intended to prepare \( trm \), \( m = 1; \ldots; |TR| \), for the search algorithm [8]. The preprocessing step involves defining a grid on \( H \cup I \) which divides \( H \cup I \) into a set of cells, called grid cells. Consequently, each \( trm \) is placed in one of the cells, which are referred to as the parent cell for that \( trm \). All the \( trm \) points that are in a cell are called child points of that cell.
In the preprocessing step, we determine each cell’s child points and store them in an array. The parent cell of each point is simply determined from the point’s coordinates. The critical parameter in the preprocessing step that affects the runtime of RBCVT-Fast is \( C_{avg} \), which must be a constant for any size of \( \text{TR} \). We have informally (empirically) observed that \( C_{avg} = 20 \) produces the most efficient algorithm with respect to runtime. Having the \( C_{avg} \) value, we can calculate the number of cells in each dimension, \( GN \), given by

\[
GN = \text{Round} \left( \frac{|T|}{C_{avg}} \right)
\]

Consequently, the total number of cells in a 2D space is \( GN \times GN \).

```
Begin
    \( L \leftarrow 0 \) // \( l \) denotes the layer number
    \( MD \leftarrow 1 \) // \( Md \) indicates minimum distance
    While dist(bj, l) < MD do
        For each cell in \( C_l \) do
            If dist(bj, cj) < MD then
                If dist(bj, \( c_l \)) < MD then
                    trwinner = dist(bj, \( c_l \))
                    MD = dist(bj, trwinner)
                    end if
                end if
            end for
        end while
    end if
End
```

Fig 2: Pseudo code for the proposed search algorithm utilized in the proposed RBCVT-Fast algorithm

### IV. A NOVEL SEARCH ALGORITHM

A novel search algorithm is discussed which reduces the linear runtime order of clustering bj to a constant runtime. The main idea behind this search algorithm is that do not need to compare bj with all of the \( \text{trm} \). As indicated in Fig. 3, to find the nearest point to bj, we need to calculate the distance between bj and the children of the adjacent cells, not all the cells. That is, we need to compare bj with the children of Cl (a set which contains all the cells in layer l), where 1 starts from zero. Layer l includes all the cells that have a similar distance from the cell with bj as a child. The highlighted cells in Figure below are in layer one.

![Fig. 3: Searching space](image)

This algorithm starts by calculating the highlighted \( \text{trwinner} = \beta(bj; clm) \) for layer zero, where each cell of Cl is denoted by clm (clm for layer zero is only one cell which is the cell parent of bj). Then, we check that \( \text{trwinner} \) is the nearest point to bj by comparing dist(bj, trwinner) with dist(bj, 1). If dist(bj, trwinner) < dist(bj, 1), then the process is finished and \( \text{trwinner} \) is the nearest point of \( \text{TR} \) to bj. Otherwise, we have to compare bj with the children of layer one’s cells and update \( \text{trwinner} \) in case we found a closer point to bj. To reduce the runtime complexity, bj is only compared with the children of those cells in layer one that dist(bj; clm) < dist(bj; trwinner). This process will continue until it finds the nearest point to bj. Pseudocode for the proposed search algorithm is indicated in Figure.

V. EXPERIMENTAL RESULT

To evaluate the proposed RBCVT approach on a testing framework which utilizes independently produced programs, this work utilizes 5 programs[21], written in Java, which implements basic mathematical functions and class objects oriented functions. In this directly utilized the source code without any modification. To evaluate the proposed RBCVT approach on a testing framework which utilizes independently produced programs, this work utilizes 5 programs, written in Java, which implements basic mathematical functions and class objects oriented functions. In this directly utilized the source code without any modification.

#### 5.1 Parameters of test case generation methods

A number of parameters are associated with each ART algorithm, which is considered constant through all the experiments. The selected the value of these parameters as recommended in their respective works. The k in FSCS method, representing the number of randomly selected candidates, is held constant at \( k = 10 \) based on the recommendation similarly, the coverage ratio in the RRT method is considered constant at 1.5. The \( k \) (population size) has been set to 20 and the probability of crossover is set at 0.6. Furthermore, the probability of mutation is considered as 0.1, the size of the mutation was set at 0.01, and the stopping criterion is set to the constant number of 100 iterations

#### 5.2 Simulation Framework

For the simulation framework, this thesis will introduce the utilized failure patterns, the failure rate associated with each failure pattern, the number of test cases in each test set, and the number of test sets.

##### 5.2.1 Failure Patterns and Failure Rates

To be able to evaluate test case generation methods, it is necessary to consider some parts of the input domain as a failure area, where a failure is produced when a test case is placed in this area. Several works have performed an empirical investigation through failure patterns within the input domain. The block pattern is generated by randomly choosing a point in ‘I’ and then a square is constructed around this point with respect to the failure rate. Due to the section of the random point near the boundaries of I, the constructed block pattern may not fit within I. In this situation, this pattern is disregarded and another random point is selected until a valid block pattern is generated. The strip pattern is generated using a random point in I and a random angle associated with...
a line passing over the selected random point. The width of the strip pattern is calculated according to the failure rate.

5.3 Performance Comparison of the Proposed Method with the Existing Approaches

Table 1: Comparison of Existing Methods before Applying Proposed RBCVT and after Applying Proposed RBCVT

<table>
<thead>
<tr>
<th>Method</th>
<th>Before the Proposed RBCVT Process</th>
<th>After the Proposed RBCVT Process</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>std</td>
</tr>
<tr>
<td>RT</td>
<td>0.017</td>
<td>0.0031</td>
</tr>
<tr>
<td>Sobol</td>
<td>0.0093</td>
<td>0.0058</td>
</tr>
<tr>
<td>FCFS</td>
<td>0.014</td>
<td>0.0023</td>
</tr>
<tr>
<td>ART</td>
<td>0.0133</td>
<td>0.0025</td>
</tr>
<tr>
<td>QRT</td>
<td>0.0438</td>
<td>0.0046</td>
</tr>
</tbody>
</table>

The mean values of the P-measures appear dissimilar for the different approaches; whereas the corresponding results after the application of the RBCVT process represents a sizable reduction of the variation between these values. Therefore, for comparison of RBCVT, as a single method, against all other approaches, we assume the average RBCVT results as the performance of the RBCVT approach. The table represents the effect size of the testing effectiveness at each strategy against RT in the block pattern simulations. Contrasting RBCVT against FSCS, RT, Sobol, ART, and QRT this highlights the increased efficiency of RBCVT regarding the lock pattern. Another conclusion from this figure is that all of the testing methods outperformed RT at every failure rate with respect to the block pattern. Comparing the amount of improvement (effect size) among all approaches in Table, one can observe that the largest RBCVT improvement belongs to the RT for all failure rates.

Fig 4: Performance Comparison of P-Measure Before and After Applying Proposed RBCVT in Random Testing With the Mean and Standard Deviation as Metrics

Fig 5: Performance Comparison of P-Measure Before and After Applying Proposed RBCVT in Sobol Testing With The Mean And Standard Deviation As Metrics

Fig 6: Performance Comparison of P-Measure Before and After Applying Proposed RBCVT in FCFS Testing With The Mean And Standard Deviation As Metrics

Fig 7: Performance Comparison of P-Measure Before and After Applying Proposed RBCVT in Adaptive Random Testing With the Mean and Standard Deviation as Metrics

Fig 8: Performance Comparison Of P-Measure Before And After Applying Proposed RBCVT In Quasi Random Testing With The Mean And Standard Deviation As Metrics
5.4 Performance Comparison Based On Improvement of Test Case Generation Methods with Respect to Different Failure Rates

Comparing the amount of improvement (effect size) among all approaches in Table 2, one can observe that the largest RBCVT improvement belongs to the RT for all failure rates. In contrast, no individual method has the smallest increase in effectiveness regarding the effect size. The table indicates the improvement of each approach after the RBCVT process comparing to the effectiveness of test cases used as inputs to the RBCVT process (effect size) with respect to block pattern at each failure rate. In this figure, in all methods, the level of changes before and after the RBCVT process is decreasing as the failure rate decreases.

Table 2: Improvement of Test Case Generation Methods With Respect To RBCVT Process at Different Failure Rates Regarding the Block Pattern

<table>
<thead>
<tr>
<th>Method</th>
<th>$10^{-3}$</th>
<th>$10^{-4}$</th>
<th>$10^{-5}$</th>
<th>$10^{-6}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>RT</td>
<td>0.44</td>
<td>0.34</td>
<td>0.26</td>
<td>0.25</td>
</tr>
<tr>
<td>Sobol</td>
<td>0.32</td>
<td>0.21</td>
<td>0.2</td>
<td>0.12</td>
</tr>
<tr>
<td>FCFS</td>
<td>0.45</td>
<td>0.18</td>
<td>0.13</td>
<td>0.09</td>
</tr>
<tr>
<td>ART</td>
<td>0.56</td>
<td>0.31</td>
<td>0.06</td>
<td>0.06</td>
</tr>
<tr>
<td>QRT</td>
<td>0.68</td>
<td>0.2</td>
<td>0.04</td>
<td>0.03</td>
</tr>
</tbody>
</table>

5.5 Performance Comparison of Existing Approaches with the Proposed RBCVT Adapter Approaches Based On the P-Measure Testing

Similar to the simulation framework results, the table and chart provides a comparison among all of the approaches where the RT effectiveness is considered as a reference, i.e., table and chart represents the effect size of each strategy against RT. In contrast with the simulation framework, the P-measure results, after the. Accordingly, in table RBCVT results with QRTs as generators are combined as QRT-RBCVT, while RBCVT with other inputs is represented separately.

Test case generation approaches in table are sorted based on their performance where the QRT-RBCVT is the approach with highest efficiency and Sobol has the worst results in term of testing efficiency. Finally, as demonstrated in Fig. 9, QRT methods revealed degraded performance compared to RT in most of the cases, whereas other test case generation approaches outperformed RT.

Table 3: P-Measure Testing Effectiveness for each Test Case Generation approach against RT with respect to the Mutants’ Framework

<table>
<thead>
<tr>
<th>Method</th>
<th>10</th>
<th>20</th>
<th>50</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sobol</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>RT</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>FCFS</td>
<td>5</td>
<td>6</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>ART</td>
<td>8</td>
<td>7</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>QRT-RBCVT</td>
<td>12</td>
<td>13</td>
<td>11</td>
<td>9</td>
</tr>
<tr>
<td>RT-RBCVT</td>
<td>15</td>
<td>16</td>
<td>13</td>
<td>10</td>
</tr>
<tr>
<td>FCFS-RBCVT</td>
<td>17</td>
<td>18</td>
<td>15</td>
<td>11</td>
</tr>
<tr>
<td>ART-RBCVT</td>
<td>19</td>
<td>18</td>
<td>16</td>
<td>13</td>
</tr>
<tr>
<td>QRT-RBCVT</td>
<td>22</td>
<td>20</td>
<td>18</td>
<td>15</td>
</tr>
</tbody>
</table>

Fig 10 Comparison of P-Measure Testing Effectiveness of each Test Case Generation approach against RT with respect to the Mutants’ Framework

VI. CONCLUSION

In this paper, the novel RBCVT (Random Border Centroidal Voronoi Tessellation) method has been proposed to the domain of software testing with the aim of increasing the effectiveness of test case generation approaches. The RBCVT method cannot be considered as an independent approach since it requires an initial set of input test cases. This method is developed as an add-on to the previous ART (Adaptive Random Testing) and QRT (Quasi Random Testing) methods, enhancing the testing effectiveness by more evenly distributing test cases across the input space. In addition, the applied probabilistic approach for RBCVT generation allows different sets of output to be produced from the same set of
inputs, which makes RBCVT an appropriate method for software testing applications.

The computational cost of a test case generation algorithm should be carefully considered in a practical application. In this research, we optimized the probabilistic computational algorithm of the RBCVT approach. The proposed search algorithm reduces the RBCVT computational complexity from a quadratic to a linear time order regarding the size of the test set, while ART methods still suffer from high runtime order. In this regard, the computational cost of RBCVT is quite feasible with respect to practical applications. It is worthwhile to state that since the RBCVT approach requires initial test cases, the computational cost of the input test set generation is added to the RBCVT calculation cost. Since the results provided indicate on average, “similar” results for RBCVT with different types of generators, we can select the RT method, which is linear and adds a low computational overhead onto the RBCVT execution. Therefore, with a concatenation of the RT and the RBCVT-Fast methods, we can produce a linear algorithm with respect to computational complexity, although in some specific situations this may lead to a slight reduction of algorithmic effectiveness.

The principle contribution of this research is utilizing CVT to develop an innovative test cases generation approach, in particular RT-RBCVT-Fast with linear order of computational complexity similar to RT.

REFERENCES


