A Note on Optimal Testcase Generation for Baundary Value Analysis

Hiroshima University GUO Xiujing
01013754 Hiroshima University OKAMURA Hiroyuki
01307065 Hiroshima University DOHI Tadashi

1. Introduction

Software testing is one of the most important activities to ensure software reliability. The main activity of software testing is to generate test cases consisting of test inputs and their expected behavior (outputs) of software. In testing phase, the test cases are executed, i.e., we obtain the actual outputs of implemented software with the test inputs, and they are compared with the expected ones so that the behavior of software is validated. To ensure the reliability of software, it is important to design the set of test cases that can find all the inherent software bugs.

There are several approaches to design test cases. The test coverage plays the most important role for the design of test cases. The test coverage is one of the criteria to measure how much the software execution paths are covered by the set of test cases. For example, the statement coverage, called C0 coverage, is defined as the percentage of the program statements executed by the test cases over all the program statements. The test cases are designed to achieve a test coverage attains a pre-defined level. On the other hand, the boundary value analysis (BVA) is also commonly used to define effective test cases. The BVA is to identify the boundaries of program paths, i.e., the values that changes program paths with a small amount of changes of inputs, and to design the test cases taking the boundary values [1]. Since it is empirically known that software bugs exist around the boundaries, the test cases with BVA are the ones capable of finding bugs with high probabilities.

In the past literature, the test case designs with test coverage and BVA are separately discussed. The main reason is that it is difficult to evaluate the test coverage intending BVA, since the boundaries are defined as like continuous values. In this paper, we attempt to define alternative measures, called boundary coverage distance (BCD), how much the test cases cover the boundaries. In addition, based on BCD, we consider the optimal test case generation to minimize BCD under the random testing scheme.

2. Boundary Coverage Distance

To generate boundary test cases, we propose a metric called Boundary Coverage Distance (BCD) to measure boundary coverage. We define BCD as the maximum distance that the test set can cover a set of boundaries and calculate the BCD according to the distance between the test case and the boundary point. The boundary point is a set of points that can represent the boundary obtained by analyzing the source code. We assume that there is a test set T containing n test cases $T = (t_1, t_2, ..., t_n)$. The process is as follows:

Step 1: Select m boundary points $P = (p_1, p_2, ..., p_m)$.

Step 2: For the boundary point p_i, calculate the distance between p_i and all test cases, and select the minimum distance p_i_min as the distance that can cover the boundary point, that is,

$$p_i\_min = \min_{j=1}^{n} dist(p_i, t_j)$$

Step 3: The maximum distance among the selected minimum distances is taken as the distance that the test set T can cover the boundary, that is,

$$BCD = \max_{i=1}^{m} p_i\_min$$

In Step 2, the dist is defined as the Euclidean distance. For example, In Fig. 1, we selected five points marked in square as boundary points. For the boundary point p_i, the test case t_2 can cover the boundary point p_i with the minimum distance.

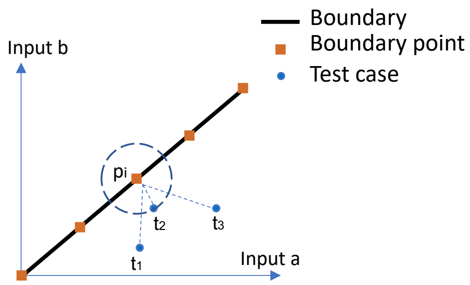

Figure 1: An example for cover a boundary point.

3. Boundary Random Testing (BRT)

BRT is a variant of the random testing schemes to generate test cases near the boundary. The main idea is to optimize a set of test cases by reducing BCD. The optimization process is shown in Algorithm 1.

During the test data generation process, the algorithm first randomly generates n test data from the input domain as an initial test set. Then we optimize the initial test set by reducing the BCD value(lines 2-11). During the optimization process, the algorithm first calculates the BCD of the initial test set. Then randomly select a test data t from the initial test set, and generate a new candidate t' according to the proposal distribution, provided that t is given $t' \leftarrow Q(t';t)$. If replacing t with candidate t' can reduce the value of BCD, the candidate t' is accepted and replaces t in the initial data set, otherwise candidate t' is rejected. After the optimization process performs a fixed number of iterations, the initial test data is moved to the boundary.

Algorithm 1 An algorithm to generate boundary test cases by reducing BCD

1: $Initial\_test\_dataset$ ←{randomly generate n test data from the input domain}
2: **while** $j \leq Iter$ **do**
3: $BCD \leftarrow BCD(Initial\_test\_dataset)$
4: $t \leftarrow randomly\ select\ a\ test\ data\ from\ Initial\_test\_dataset$
5: $t' \leftarrow Q(t';t)$
6: $Candidate\_test\_dataset$ ← {Replace the t in the Initial_test_dataset with the candidate t'}
7: $BCD' \leftarrow BCD(candidate\_test\_dataset)$
8: **if** $BCD' < BCD$ **then**
9: $Initial\_test\_dataset \leftarrow candidate\_test\_dataset$
10: **end if**
11: $j \leftarrow j + 1$
12: **end while**

4. Experiment

We conducted an experiment to generate test cases for a simple C program for judging whether the English test passed. We assume that an English test is divided into two parts: listening test and reading test. The full score of each test is 100.0 points. When the score of each test is greater than or equal to 50.0 points, and the sum of the scores of the two tests is greater than or equal to 120.0 points, Pass the English test. This C program has seven boundaries and two floating inputs a and b. And we observe the distribution of test cases generated by BRT.

In the experiments, the RT randomly generates $n = 20$ test cases from input domain according to the uniform distribution. Then we use BRT to optimize the RT generated test set. We use the normal distribution as the proposal distribution in BRT, and the candidate is a random sample drawn from the normal distribution with the current sample x as the mean and a variance $v = 100$. Then we select 13 boundary points, as shown in Fig. 2.

Fig. 2 shows the data distribution generated by RT and BRT. It can be seen intuitively from the distribution graph that the BRT method can generate test cases near the boundary.

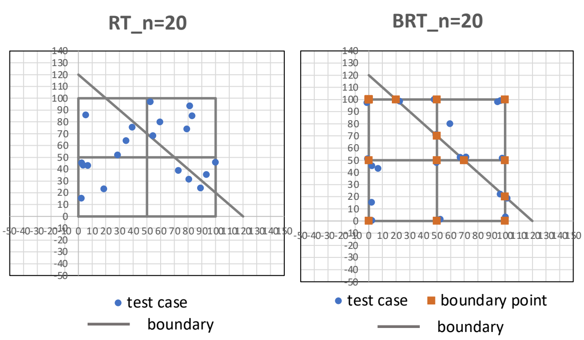

Figure 2: The data distribution generated by RT and BRT.

References

[1] Reid S C. An empirical analysis of equivalence partitioning, boundary value analysis and random testing[C]//Proceedings Fourth International Software Metrics Symposium. IEEE, 1997: 64-73.