Equivalence Class Testing

Anuruddha I Jayasekara Pathiranage

01/29/2015

Equivalence Class Testing

Equivalence class theory is proposed by Glenford Myers. Equivalent Class Partitioning is a black box technique that can be applied to all levels of testing such as unit, integration, system, etc. It attempts to reduce the total number of test cases necessary by partitioning the input conditions into equivalence classes. The principal of equivalence class testing is formation of equivalence classes. In this method, input or output data is grouped into sets of data that behave in a same way using an Equivalence relation.(Gupta, n.d.) In this method we consider both valid and invalid input domains. An equivalence relation describes how data is going to be processed when it enters a function. In order to get the full potential of the equivalence class testing, some time should be spent on designing a strong equivalence relation. Creating a strong equivalence relation will result in optimal and useful test cases.(“LU-raksti-751.pdf,” n.d.)

Example of Equivalence Class Testing

For a job, male and female employees are accepted in age range of 21-40.

Here, we have two equivalence classes of valid input data.

Field	Description
1. Age	≥ 21 and ≤ 40
2. Sex	Male and Female

Following are the classes of invalid input.

Field	Description
3. Age	Empty

4. Age	<21
5. Age	>40
6. Sex	Neither male nor female

It is important to pick one invalid value per test case in order to reduce missed product bugs.

Using the above equivalence classes, we can design following tests.

- Valid Age from #1 and valid Sex from #2
- Valid Age from #1 and invalid Sex from #6
- Invalid Age from #3 and valid Sex from #2
- Invalid Age from #4 and valid Sex from #2
- Invalid Age from #5 and valid Sex from #2

When Equivalence Class Testing should be used

Equivalence class testing is appropriate in situations like;

1. When the program function is complex. That means if the system has number of test items that need to be tested, Equivalence class testing is useful. If we test each of test item in regular manner, it waste both time and money.
2. When there is strong need to avoid redundancy.

In these situations, we group the test items in to classes where all items in the particular class supposed to behave similarly. Then we only need to test one item from each class, to ensure that the system works properly.(Kuhn, Wallace, & Gallo, 2004)

Types of software testing problems that Equivalence class testing addresses

- For triangle problems

The program takes three input numbers a, b and c that represent the lengths of the three sides of a triangle. Based on these three input values, the program determines whether the triangle is scalene (that is, it has three unequal sides), isosceles (two equal sides), or equilateral (three equal sides).

1= {the triangle is equilateral}

2= {the triangle is isosceles}

3= {the triangle is scalene}

4= {sides a, b and c do not form a triangle}

Test case	a	b	c	Expected output
WN1	5	5	5	Equilateral
WN2	2	2	3	Isosceles
WN3	3	4	5	Scalene

WN4	4	1	2	Not a triangle

- For NextDate functions

Nextdate is a function of three variables date, month and year. These have intervals of valid values as follows:

M1 = { month :1 < month <12}

D1 = {day :1< day <31}

Y1 = { year :2000 < month <2016}

The invalid equivalence classes are

M2= {month: month <1}

M3= {month: month >12}

D2= {day: day <1}

D3= {day: day >31}

Y2= {year: year <2000}

Y3= {year:year>2016}

Key assumptions of Equivalence class testing

1. Program is a function from Input to Output
2. Input and/or output variables have well-defined intervals

Limitations of Equivalence class testing

1) Several tries should be required to find the correct equivalence relation.
2) Does not work well for Boolean variables
3) Does not work well for logical variables
4) Not useful when variables are not independent
5) Not useful for strongly typed languages
6) Difficult to determine expected values for invalid variable values(Copeland, 2004)

Reference

Copeland, L. (2004). *A Practitioner's Guide to Software Test Design*. Artech House.

Gupta, Y. N. (n.d.). Equivalence Class Testing-Black Box Software Testing Techniques. Retrieved January 30, 2017, from http://www.softwaretestinggenius.com/equivalence-class-testing-black-box-software-testing-techniques

Kuhn, D. R., Wallace, D. R., & Gallo, A. M. (2004). Software fault interactions and implications for software testing. *IEEE Transactions on Software Engineering*, *30*(6), 418–421. https://doi.org/10.1109/TSE.2004.24

LU-raksti-751.pdf. (n.d.). Retrieved from

http://dspace.lu.lv/dspace/bitstream/handle/7/1301/LU-raksti-751.pdf?sequence=1&isAll

owed=y#page=80