A Novel Approach for Controlling a Size of a Test Suite with Simple Technique

B.Galeebathullah
Student,Department of CSE,
Anna UniversityTiruchirappalli,India.
Email-kalifathullah@gmail.com

C.P.Indumathi
Lecturer,Department of CSE
Anna University,Tiruchirappalli,India
Email:inducp@gmail.com

`***Abstract***-*Software testing is an important activity in the software development life cycle. and also expensive phase when compared to all other phases of the software development life cycle. Software testing purpose is to detect,software failures so that defects may be recoverd and corrected.Softwares are tested through test cases.A test case is a set of condition or variables variables under which a tester will determine whether an application or software system is working correctly or not.For many application test cases are generated automatically but the main problem is selecting a effective test cases among all test cases.The process of selecting a effective test cases from all test cases is called Test case selection.This process is used for finding a redundant test cases and removing the redundant test cases in a test suite is called test suite minimization. In this paper ,we propose a novel test suite reduction based on set theory.To demonstarate the applicability of this approach ,we conduct an experimental study ,The result shows that our technique is easy to implement and consumes less time*

Categories and Subject Descriptors: [Software Engineering]:
Testing and Debugging

Keywords : software testing, test case minimization, test suite minimization, set theory,test suite minimization

1 INTRODUCTION

Software testing is the process of executing a program or system with the intent of finding errors. As a part of any software development process, software testing represents an opportunity to deliver quality software and to substantially reduce development cost as much as 50% [1].
As software testing is an expensive software development activity,the cost of testing to achieve certain adequency according to given criterion is also importance comparison of testing costs involves many factors ,one of the simplified measure of test cost is the size of an adequacy criterion.

Generally softwares are tested through a test case.A test case is defined in IEEE standard as [13]:" A set of inputs, execution, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement" **.** The quality of a test cases measured is based on the following factors: 1) Code Coverage, 2) fault Coverage, 3) size, 4) the number of faults detected by the most effective test contain [12]. A test case is usually a single step, or occasionally a sequence of steps,to test the correct bahaviour. Test cases are reffered to as test scripts,when particularly written, written test cases are usually collected into test suites.A test suite often contains detailed instructions or goals for each collection of test cases and information on the system configuration to be used during testing. A group of test cases may also contain prerequisite states or steps, and descriptions of the tests. As one test case can hardly satisfy all the requirements, it is usually required to use a suite of test cases to satisfy as many as possible requirements. Intuitively, the more test cases are used, the more possible the requirements are satisfied. Practically, a test suite usually undergoes a process of expansion, as new test cases are inserted into the test suite to ensure the requirements being satisfied. As a result, a test suite may contain more than enough test cases for satisfying the requirements.Therefore ,controlling the size of a test suite based on the requirements by keeping a minimum test suite size is called test suite reduction or test suite minimization.

2 RELATED WORK

A classical greedy heuristics [2][3] solves the problem by recursive action of the following steps.select the test cases from a test suite and find out wheather test cases satisfy the most requirements and remove the requirements covered by the test cases.This process will be stop until all the requirements are coverd or satisfied by the test case.Another heuristic developed by Harrold Soffa and Gupta[4] (HGS):Selects a representative set of test cases from a test suite until covered a requitements by a test cases.This approach will take additional computation for selecting a test cases.However the above two approaches we can't exactly said that the which one gives better reduction of the test cases.The main drawbacks in existing approach is that they lack of fault detection capability.Wong et al[5][6] reported that all uses

coverage was kept constant ,test suites could be minimal at line or no cost in fault detection effectiveness. Von Ronne [13] generalized the HGS algorithm, such that every requirement could be satisfied multiple times according to its hitting-factor Rothermal[7][8] expressed that the fault detection capabilities of test suites can be nearly equal to the minimization of the test suites. Tallam and Gupta[14] developed another heuristic called the delayed-greedy strategy.Concept analysis is a hierarchical clustering technique for objects and their corresponding attributes. When viewing test cases as objects and requirements as attributes, the framework can help expose both the implications among test cases and the implicationsamong those requirements satisfied by the test cases. In their experiments, the delayed-greedy strategy consistently obtained. the same or more reduction in suite sizes than it did in prior heuristics, such as in the HGS or in the classical greedy strategy.

The remainder of this article is organized as follows.In section 2 we review the test suite reduction problem,the existing solutions.The implementation of the proposed approach is described in section 3.Finally,the conclusion and future work are given in section 4.

2 TEST SUITE MINIMIZATION

In this section, we review the definition of the test suite reduction problems ,the existing algorithms

2 .1 BACKGROUND AND DEFINITION

According to Rothermal [9],a Software contains 20,000 LOC (Lines of Codes) requires seven weeks to run all the test cases.to eliminate a redundant test cases in a test suite,test suite minimization approach is necessary.The test suite minimization definition as follows .A test suite consists T test cases {t1,t2,t3……..t_n} satisfy the requitements {r1,r2,r3,……..r_n} that must be coverage exists among the requirements and test cases of a progam.to finding a minimum subset of t that will covers a maximum coverage requirement.

Problem:To find a subset of Test case T that will capable to cover the maximum requirements by test suite T from an unminimized test suite.

2.2 EXISTING SOLUTION

Table 1 is an example that shows the coverage information between the test cases in a test suite {t1,t2,t3,t4,t5}andrequirements{r1,r2,r3,r4,r5,r6},here the requirements denoted any one of the coverage criterion such as statement,Branch so on .The symbol X denotes the coverage between the test cases and requirements.

Test case/Requirements	T0	T1	T2	T3	T4	T5
Ro	X	X	X			X
r1	X	X			X	
r2				X		
r3	X				X	
r4	X			X	X	
r5		X	X	X		X
r6		X	X	X		

Table -1

The Classical Greedy heuristic provides a solution for set-covering problem [2][3] for finding a minimization of test cases from unminimized test suite .The Algorithms woks as follows:select the test cases that meets the maximum uncovered requirements until all the requirements are covered.from the test case.the reduction of the test suite is $T_1.T_3,T_4$.The HGS heuristic as Follows First find out the association beteeen the test cases and requirements and find representative set that covers all the requirements .It consider a $T_i$of the single element (Cardinality one) then places in a test suite .Next select the cardinality two are considered until all the requirements are satisfied by the test case.the reduced test suite for HGS is $T_{0,}T_3$

3.PROPOSED TEST CASE REDUCTION TECHNIQUE

In this section first we describe about set theory and its operation, next we will describe how the set theory is applying in the test suite reduction.The concept of set theory is fundamental to mathematics and computer science. Everything mathematical starts with sets.for example, relationships between two objects are represented as a set of ordered pairs of objects.

Definition 1:**Equiality of Sets**: Two sets are equal if and only if they have the same elements. For any sets A and B ,A=B if and only if $\forall$ x [x $\in$ A $\leftrightarrow$ x $\in$ B]

Definition 2: Subset :A set ***A*** is a subset of a set ***B*** if and only if everything in ***A*** is also in ***B***. More formally, for any sets ***A*** and ***B***, ***A*** is a subset of ***B***, and denoted by A $\subseteq$ B if and only if $\forall$ x [x $\in$ A $\leftrightarrow$ x $\in$ B]

In our approach we used the set theory from that we find the intersection between the one requirements to another reruiremets of branch coverage criteria for the set of test cases.the working procedure of a test case reduction algorithm is as follows:

Input:
Set of requitements (R):{$r_1,r_2,r_3,\ldots\ldots r_k$}
Set of Test cases (TC):{$t_1,t_2,t_3\ldots\ldots\ldots t_n$}
Output: Selected:minimized test suites

Step1:
Find the intersection of the one requitrements to another requirements

Step2:
If any intersection elements occur then added to the reduced suite

Step3:
Repeat the process until all the requirements are satisfied by the test cases .

Algorithm Test suite reduction

Input:
Set of requirements (R) : {$r_1,r_2,r_3,\ldots\ldots r_k$}
Set of Test cases (TC) : {$t_1,t_2,t_3\ldots\ldots\ldots t_n$}
T_i : set of test cases satisfying requirements r_i ,i=1,2,…..k
R_i : Set of requirements satisfying test cases t_i, i=1,2,……..n

Output: Selected test cases :S
Declare
$r_{i=0;}$
$r_k = n;$
list =0;

Begin

S ← { }//initially selected test case is empty
foreach($r_{i=0;}$ TO $r_k = n$)
step 2: if ($r_i \cap_i r_k$)
{
S ← { t_i } // added to the selected test cases
}
else
goto step 2:
forend
list ← $\cup_i t_i$ // reduced test suites

3.1 MEASURE

In this paper, we use the formula finding a size of a reduced test suite

- The percentage of test suite size reduction (SSR)[10] is defined as

$$SSR = T - T_{min} / T \quad \text{---------- (1)}$$

Where T is the number of test cases in a orginal test suite.T_{min} is the number of test cases in the reduced suite/minimized suite. A higher SSR means a better reduction .

4.EXPERIMENTAL STUDY

In section 3 ,we described a proposed approach based on the set theory concepts.for the purpose of understanding In this section we do experimental on a small program based on the branch coverage criterion.we choose a small program taken from[16] .

```
int foo (int a,int b,int c)
{
 B1:      if( a>0)  x=x+1;
           else      x=x-1;
 B2:       if(b>0)  y= -2;
          else      y=x+1;
 B3:       if(c>0)
        {
 B4:         if(y<0)  return 10;
        }
           else
            return 20;
```

Fig-2 A Simple Program based on Branch Coverage

Test case	A	B	C
T1	1	-1	1
T2	-1	1	1
T3	1	1	-1
T4	-1	1	-1
T5	-1	-1	-1

Table -2 Test cases

Table 2 shows the developed test cases inputs for the sample program mention in the fig 2 and Table 3 displays the corresponding branch coverage matrix

between the test cases and the requirements covered by the sample program

Branch/ Test case	Cardinality	T1	T2	T3	T4	T5
B1	2	X		X		
B2	3		X		X	X
B3	3		X	X	X	
B4	2	X				X
B5	2	X	X			
B6	3			X	X	X
B7	1		X			
B8	1	X				

Table-3 Test Case Coverage Matrix

In our experiment we use a coverage criterion as branch coverage ,by applying this coverage matrix in to our algorithm we find out the intersection between the requirements of r_i to r_k until all the requirements are satisfied.intially the selected test suite are empty.the resulted selected test cases for our approach is t_1, t_2, t_4

4.1 EXPERIMENTAL RESULTS

In this section we describe the average sizes of the reduced suites based on the Greedy,HGS,Set theory for the above program.The table 4 shows the percentage of test suite size reduction (SSR) for the above program.

Algorithm	Orginal Test case	Reduced Test case	% of Reduced Test suite
Greedy	5	3	57.14
HGS	5	3	57.14
Set theory	5	3	57.14

Table 4-Percentage of a reduced test suite

5.CONCLUSION AND FUTURE WORK

In this paper we presented a new algorithm to controlling a size of a test suite that covers all the requirements covered by a test suites.our techniques is simple among all other techniques.In future we would like to runs similar experiments on programs from a broader range of programming languages,sizes and problem domains ,in order to capture how large programs can quickly lead to test suite reduction

REFERENCES

[1] Boris Beizer,"Software Testing Technique", Second Edition International Thomson Computer 1990 ISBN 1-85032-880-3

[2] T.H.Cormen, C.E.Leiserson, R.L.Rivest, "Introduction to Algorithms,MIT Press,Cambridge,MA 2001."

[3] V.Chivtal, A Greedy Heuristic for the set covering Problem, Mathematics of Operation Research 4 (3) (1979) 233-235

[4] M.J.Harrold, R.Gupta, M.L.Sofa,"A Methodology for controlling the size of a test suite,ACM Transcations on Software Engineering and Methodology 2(3)(1993) 270-285

[5] W.E.Wong,J.R.Horgan, S.London, A.P.Mathur,"Effect of test set minimization on fault detection effectiveness,Software-Practice and Experience 28 (4) (1998) 347-369.

[6] W.E.Wong,J.R.Horgan, S.London, A.P.Mathur,"Effect of test set minimization on fault detection effectiveness,in ":Proceddings in Seventeeth international conference on Software Engineering ,Seattle,Washington,USA,1995,pp 41-50

[7] G.Rothermal,M.J.Harrold,J.Von Ronne,C.Hong,"Em-perical studies of Test suite reduction",Software testing verification and relaiblity 12 (4) (2002)219-249

[8] G.Rothermal,M.J.Harrold,J.Ostrin,C.Hong,"An Empeerical study of the effects of minimization on the fault detection capabilities of test suites ",Proceedings of The fourteenth international conference on software maintainance ,Betheseda,MD,USA,1998 ,pp 34-43

[9] G.Rothermal,R.H.Untch,C.Chu,M.J.Harrold,"Prioritizing test cases for regression testing",IEEE Transcations on software Engineering ,27 (10)(2001) 929-948

[10] Dennis Jeffrey and Neelam Gupta "Improving Fault Detection Capability by Selectively Retaining Test Cases during test suite reduction ",2007,pp 1108-123

[11] H.Agrawal,"Efficient Coverage Testing Using Dominotor Graphs," Workshop on Program Analysis For Software Tools and Engineering ,1999,pp 11-20

[12] http://www.cs.odu.edu/~toida/nerzic/content/set/basics.html

[13] I. Pomeranz, S.M. Reddy, "On the Compaction of Test Sets produced by Genetic optimization ",In proceedings of IEEE 5th Asian Test Symposium (ATS'97), Akita, Japan, November,1997

[14] J.V. Ronne, "Test Suite Minimization: An Empirical Investigation ",Bachelor Thesis, June 1999, Retrived From url http://www. http://www.ics.uci.edu

[15] S.Tallam, N.Gupta, A Concept Analysis inspired greedy algorithm for test suite minimization,in:Proceedings of the Sixth Workshop Program Analysis for software Tools and Engineering, Lisbon, Portugal,2005

[16] Jun-wei Lin , Chin-Yu Hang," Analysis of test suite reduction with enhanced tie-breaking",in:information and software technology 51 (2009) 679-690