Reduction of Test Cases and Prioritization by using Weight Concept

Tarun Kumar Gautam, Neha Bajpai

Abstract*—Length and complexity of the software are rising day by day. So software testing is particularly difficult for development of high assurance software such as software that is produced commercial airborne system. Modified condition/Decision Condition (MC/DC) is an effective verification technique and can help to remove safety faults. This paper present new approach for test case reduction and prioritization that effectively use MC/DC criteria. In this approach we used weight concept for discarding weak test case and set priority.

***Keywords*— Test case reduction, Truth vector, Prioritization, Modified condition/Decision condition, coverage**

I. INTRODUCTION

Software Testing is the process of executing a program with the intent of finding errors [1]. Software testing can be considered as combination of validation and verification activity. Software testing is a very important activity in software development life cycle. It is one of the most promising way to ensure quality of the developed software. Software testing consumes nearly 50% of the total development cost of the software. One cannot do exhaustive testing under project deadline because it requires lots of effort and time. Thus, to limit the process of testing, tester should know which test cases are effective for finding error quickly.

Testing is particularly expensive for development of high assurance software such software that is produced commercial airborne system. In initial version of software which is developed of test cases, is difficult. Updating in assurance software produces a big problem for reuse of test case. Researchers have investigated two approaches for addressing the test-suite size problem that maintain the same coverage as the original test suite: Existing test-suite reduction and prioritization techniques, however, may not be effective in reducing or prioritizing MC/DC-adequate test suites because they do not consider the complexity of the criteria. To satisfy MC/DC criteria following condition, during testing all of below must true at least once [1]

1. Each decision tries every possible outcome.
2. Each condition in decision takes on every possible outcome
3. Each entry and exit is invoked
4. Each condition in decision is showing to independently effect of decision

Independence of a condition is shown by proving that only one condition changes at a time.

Manuscript received April 2013.

Mr. Tarun Kumar Gautam received his B.Tech in CSE from Gurukul Kangri Vishvavidhayala, Haridwar, Uttarakhand, India in 2009.Currently, he is doing M.Tech in IT from C-DAC Noida(Affiliated to G.G.S.I.P.U New Delhi), India.

Ms. Neha Bajpai received M.Tech in Information Technology from the VinayakaMission University of Tamilnadu in the year 2005.

if ((A&&B)!!C)

Let A, B and C is conditions

A(T)	A(F)
TTT →T	FTT →T
TTF →T	FTF →F

In this paper we have presented a new approach for test case generation using MC/DC criteria with weight concept. To run all MC/DC truth vector is very difficult and time consuming process. Every test case keeps some weight in test suit which may be more or less. Weight is directly proportional to priority. More weight more priority. In this approach we are using weight concept to set priority of test case and discard weak test case by which we can save significant amount of time and reduce Test case.

Coverage Criteria[2]	**1**	**2**	**3**	**4**	**5**
Every statement in the program has been invoked at least once	Y	-	-	-	-
Every point of entry and exit in the program has been invoked at least Once	-	Y	Y	Y	Y
Every decision in the program has taken all possible outcomes at least Once	-	Y	-	Y	Y
Every condition in a decision in the program has taken all possible outcomes at least once	-	-	Y	-	Y

1: Statement coverage

2: Decision Coverage

3: Condition Coverage

4: Condition/Decision Coverage

5: Modified Condition/Decision Coverage (MC/DC)

Y: Yes

Let if ((A&&B)!!C)

Independent pairs for condition A

A(T)	A(F)	Independent pair
TTT→T	FTT→T	NO
TTF→T	FTF→F	YES
TFT→T	FFT→T	NO
TFF→F	FFF→F	YES

Independent pairs for condition B

B(T)	B(F)	Independent pair
TTT→T	TFT→T	NO
TTF→T	TFF→F	YES
FTT→T	FFT→T	NO
FTF→F	FFF→F	NO

Independent pairs for condition C

C(T)	C(F)	Independent pair
TTT→T	TTF→T	NO
FFT→T	TFF→F	YES
FTT→T	FTF→F	YES
FFT→T	FFF→F	YES

MC/DC Pair:

A(T)	A(F)
TTF	FTF
TFF	FFF

B(T)	B(F)
TTF	TFF

C(T)	C(F)
TFT	TFF
FTT	FTF
FFT	FFF

II. PROPOSED APPROACH

Proposed scheme consists of four steps.

Step1. Generate independent pair from requirement table

MC/DC Pair criteria in which decision has been show to independent effect that's decision outcome. Combination of two truth vector, one is true and one is false that together demonstrate independent of condition in Boolean expression. The condition will change value between two truth vectors and all other condition either will not change.

Recruitment table for gate OR Gate: Three variables

if (A || B || C)

Symbol and notation	
T	True
F	False
A(T)	Condition A Regarding True value
A(F)	Condition A Regarding False value

A	B	C	OUTPUT
T	F	F	T
F	F	F	F
F	T	F	T
F	F	F	F
F	F	T	T
F	F	F	F

Similarly for two variables

If (A || B)

A	B	OUTPUT
F	F	F
T	F	T
F	T	T

Recruitment table for gate AND Gate: Three variables

if (A && B && C)

A	B	C	OUTPUT
T	T	T	T
F	T	T	F
T	T	T	T
T	F	T	F
T	T	T	T
T	T	F	F

Similarly

If (A && B)

A	B	OUTPUT
T	T	T
T	F	F
F	T	F

III. IMPLEMENTATION

Let 'P' be program.

Program P:

{ .

.

.

```
condition1: if ((A&&B) ||C)
            {
             .
             .
             .
             }

    condition1: if (…….)
            {
             .
             .
             .
             }
            .
            .
            .
            .
}
```

Where: A, B and C are conditions in program

if ((A&&B)||C)
By using Requirement table:

Test case	A	B	C	OUTPUT
T1	T	F	F	F
T2	F	T	F	F
T3	T	T	F	T
T4	T	F	T	T
T5	F	T	T	T

Step 2: Weight of test case in independent pair
T1=TFF= A (T) +B (F) +C (F) = 1/2 + 1 + 1/3 = 1.8
T2= FTT =A (F) +C (F) =1/2 +1/3 =0.8
T3= TTF =A (T) +B (T) = 1/2 + 1 +1.5
T4= TFT = C (T) = 1/3 =0.3
T5= FTT = C (T) = 1/3 =0.3

Step3: Prioritization

Test case	T4,T5	T2	T3	T2
Test input	TFF,FTT	FTF	TTF	TFF
weight	0.3	0.8	1.5	1.8
Output	T,T	F	T	F

Step4: Discard weak test case:

Takes different combination of condition for finding maximum condition coverage
Case no1: if we discard test case T4

T5; FTT → T
T2: FTF→ F
T3: TTF →T
T2: TFF →F

Condition A:
T2: FTF→F
T3: TiTF→T

Condition B
T2: TFF →F
T3: TTF→T

Condition C
T2: TFF →F
T5: FTT→T

$$\text{CondCov} = \frac{E_T(\text{II}) + E_F(\text{II})}{2C} \times 100\% \quad [4]$$

= ((3+3)/(2*3))*100= 100%

Case no2: if we discard test case T5

T1: TFF→ T
T2: FTF→ F
T3: TTF→T
T2: TFF →F

Condition A

T4: TFF→T
T2: FTF→ F

Condition B
T4: TFF→ T

T2: FTF→ F

Condition C
T2: FTF→ F
For condition C have no test case to test True input

Condicov = ((2+3)/(2*3))*100% = 99.99 %

So for maximum condition coverage T4 Discard

Step5: Final test case:

Test case	T5	T2	T3	T2
Test input	FTT	FTF	TTF	TFF
weight	0.3	0.8	1.5	1.8
Output	T	F	T	F

Step6: Finding Coverage

Condition Coverage
Condicov = 100 %

Decision coverage [4]
Divcov = D (Outcome Test Cases)/ D (Total test cases) = (4/8).100% = 50 %

Condition-decision coverage [4]
Cov-div = (condicov + divcov)/2 = (100+50)/2 =75%

IV. EMPIRICAL STUDY

Expression	No of condition	Total no of test case	No of test case after applying reduction approach	Condition coverage	Decision coverage	Condition decision coverage
if((A&&B)‖C)	3	8	4	100%	50%	75%
if((A‖B)&&C)	3	8	4	100%	50%	75%
if((A‖B)‖C)	3	8	4	100%	50%	75%
If((A&&B) ‖ (C&&D))	4	16	8	100%	50%	75%

V. CONCLUSION

This paper has presented a new approach for test case generation using MC/DC criteria. With weight concept, group is defined according to the weightage of test cases. To run all MC/DC truth vector is very difficult and time consuming process. Every test case keeps some weightage in test suite. This paper used weight of test case and set the priority and discards weak test cases. Which significant saving of time for testing, one test case will be chosen from every group and discard other. This paper present also coverage of conditions, decision and condition-decision. This approach cover 100% condition coverage and 50% decision coverage and 75% condition-decision coverage.

For maximum condition coverage ,one criteria should be follow that no of true test case and no false test case should be equal(n(T)= n(F)).

REFERENCES

[1] http://en.wikipedia.org/wiki/Modified_condition/decision_coverage

[2] *Kelly J. Hayhurst, Dan S. Veerhusen, John J. Chilenski, Leanna K. Rierson,* "A Practical Tutorial on Modified Condition Decision Coverage".

[3] James A. Jones and Mary Jean Harrold, Member, IEEE Computer Society," Test-Suite Reduction and Prioritization for Modified Condition/Decision Coverage", IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 29, NO. 3, MARCH 2003.

[4] Venezia Elodie," White Box Coverage and Control Flow Graphs", June 21, 2011

[5] Roger S. Pressman, "Software Engineering – A Practitioner's Approach", Fifth Edition, McGraw-Hill International Edition, New Delhi, 2001.

[6] Boris Beizer, "Black-Box Testing – Techniques for Functional Testing of Software and Systems", John Wiley & Sons Inc., New York, 1995.

[7] http://www.bullseye.com/download.html

[8] "An Investigation of Three Forms of the Modified Condition Decision Coverage (MCDC) Criterion", April 2001 Final Report, U.S. Department of Transportation Federal Aviation Administration

Mr. Tarun Kumar Gautam received his B.Tech in CSE from Gurukul Kangri Vishvavidhayala, Haridwar, Uttarakhand, India in 2009.Currently, he is doing M.Tech in IT from C-DAC Noida(Affiliated to G.G.S.I.P.U New Delhi), India. He is working on the project "**Reduction of Test Cases**". His interest areas are Software Testing, Operating Systems, and DBMS.

Ms. NEHA BAJPAI received M.Tech in Information Technology from the VinayakaMission University of Tamilnadu in the year 2005. She has eleven years of teaching and one year of IT implementation experience. Presently, she is working as a Senior Faculty in School of IT at Centre for Development of Advanced Computing (CDAC), Noida.