Appl. Math. Inf. Sci. 6 No. 1S pp. 31S-34S (2012)

Applied Mathematics & Information Sciences
An International Journal

Model-Based Boundary Coverage Criteria for Logic Expressions

Liping Li[1, 2, 3] and Huaikou Miao[1]
[1] School of Computer Engineering and Science, Shanghai University, 200072 Shanghai, China
[2] Shanghai Key Laboratory of Computer Software Evaluating &Testing, 201112 Shanghai, China
[3]Computer and Information Institute, Shanghai Second Polytechnic University, 201209 Shanghai, China
Corresponding author: Email: llping2000@yahoo.com.cn

Received June 22, 2010; Revised March 21, 2011; Accepted 11 June 2011
Published online: 1 January 2012

Abstract: Boundary is especially fault-prone to system. Aim at the model-based logic coverage criteria with little regard to the boundary, this paper proposes a series of model-based logical boundary coverage criteria. These criteria are used to generate test cases automatically. Results show that test cases satisfying these criteria can detect more errors than original logic coverage criteria. They not only satisfy the logic coverage criteria but also test the system boundaries.

Keywords: Boundary Testing, Logical expression, Predicate, Test Case, Test Criteria

1 Introduction

Model-based testing is very promising for automated functional software testing. Logical expressions can be easily processed automatically. In the past, test criteria based on logical expressions did not focus on the boundary. Boundary value testing is only known as an effective heuristics for test cases generation. This paper takes advantage of the logic coverage criteria and the boundary coverage criteria, proposes a series of model-based logical boundary coverage criteria.

2 Logical Expression

Predicate is the main element of the logical expression. A *predicate* is an expression that evaluates to a boolean value, such as, ((X > Y)∧ ¬A) ∨ (X≤Z) ∨ F(x). Predicates may include boolean variables, non-boolean variables which are compared with relational operators, functions that return a boolean value, all three of which may be linked with logical operators [2].

A *literal* is a predicate that does not contain any of the logical operators or the negative of the literal. For example, the predicate ((X > Y)□¬A)∨ (X≤Z) ∨ F(x) contains four literals: (X > Y), ¬A, (X≤Z), F(x). Some documents defined them as clause; In order to consistent with the classic first order predicate logic, we define them as the literal [3].

Logical expression is the main form to describe pre- and post-condition of the formal specification. They can be used as the main object of the model-based test cases generation and the basis for the definition of test adequacy. Let P be a set of predicates and L be the set of literals in P. For each predicate $p \in P$, let L_p be the set of literals in p, that is, $L_p = \{c | c \in p\}$. Typically, L is the union of the literals in each predicate in P, namely $L = \bigcup_{p \in P} L_p$ [2]. The logical expression criteria can refer to paper [2].

3 Basic Boundary Coverage Criteria

Boundary is particularly easy to lead to system failure. At present, the boundary/domain test is generally used as the basis for test generation algorithm. The boundary values of a predicate are defined by its domain [1]. This paper considers the

system state is uniquely determined by n (state and input) variables $\{x_1, x_2,\ldots, x_n\}$. Each input variable x_i has its corresponding range D_i. The product of all variables range constitute the domain D, i.e., $D = D_1 \rightarrow D_2 \rightarrow \ldots \rightarrow D_n$. Let each D_i be bounded, so, the system reachable state space is a subset of domain D. We first introduce the formal boundary coverage criteria, definition 1 to 5 which comes from reference [1], and some changes are made in order to meet our needs.

Definition 1 (Boundary State) *Assume $s=(x_1, x_2,\ldots,x_n)$ is a state of domain D, $D \Updownarrow \measuredangle^n$, the neighborhood of s is $N(s)=\{s, (x_1 \quad \varepsilon, x_2,\ldots,x_n), (x_1, x_2 \quad \varepsilon,\ldots,x_n),\ldots, (x_1, x_2,\ldots,x_n \quad \varepsilon)\}$, $\varepsilon > 0$ is a extreme small value. We consider s is a state of D iff N(s) contains at least a point of $\measuredangle^n \setminus D$.*

Definition 2 (Domain Boundary, Br(D)) *The boundary of domain D is the set of its boundary state. For some neighborhood of s are in D, some are outside of D.*

Definition 3 (One Boundary Coverage, OBC)*TS(Test Suite) satisfies OBC on domain D iff there is at least one boundary state $s \in Br(D)$, TS contains test case making s be tested.*

Definition 4 (All Boundary Coverage, ABC) *TS satisfies ABC on domain D iff each boundary state $s \in Br(D)$, TS contains test case making s be tested.*

Definition 5 (Multi-Dimensional Boundary Coverage, MDBC) *TS satisfies MDBC on domain D iff there is some boundary state $s \in Br(D)$, TS contains test cases making every variable $x_1, x_2,\ldots,x_n$ takes its minimum and maximum on D.*

ABC is too strong to meet for a large system. It means, we must find all the boundary state of D, and then to obtain and perform a lot of test cases. OBC is the weakest criteria.

4 Logic Boundary Coverage Criteria

Logic coverage criteria are mainly used for formal specification-based testing [3]. It generates test cases by analyzing the predicates and literal truth value relationship. The formal specification is constituted by a series of states and transitions. Long-term practice shows that the system boundary is most likely to make system wrong. Boundary state is a state that at least one of those state variables can be taken to the extreme value of its sub-domain state. Aim at the logic coverage criteria with little regard to the boundary, this paper proposes a series of logical boundary coverage criteria based on paper [1] and [2].

Definition 6 (Literal Boundary Coverage, LBC) *TS satisfies LBC iff for each predicate $p \in P$, TS contains at least two test cases making at least one literal boundary in L_p be tested and c_i evaluates to true, and c_i evaluates to false.*

For the predicate $p=(a\geq 3)\vee (b < 4)$, test suit {a=3 ∧ b=3, a=2∧ b=3} satisfies the LBC.

Definition 7 (Predicate Boundary Coverage, PBC) *TS satisfies PBC iff for each predicate $p \in P$, TS contains at least two test cases making at least one literal boundary in L_p be tested and p be evaluated to true, and p be evaluated to false. This literal determines the value of predicate p.*

For the predicate $p=(a\geq 3)\vee (b < 4)$, test suit {a=3 ∧ b=3, a=2∧ b=3} satisfies the LBC but not PBC. To satisfy the PBC, first of all, we need to find the truth value combination of literal that make p evaluate to true and false. Test suit {a=3∧ b=4, a=2 ∧ b=4} satisfies both the LBC and PBC.

Paper [2] gives the definition of *Determination*. Given a literal c_i in predicate p, if the remaining minor literals $c_i \in p$, $j \neq i$, have values so that changing the truth value of c_i changes the truth value of p, then we say that c_i determines p, called c_i the major literal, c_j the minor literal. Using the literal and predicate relationship, we can define active literal boundary coverage as follows.

Definition 8 (Active Literal Boundary Coverage, ALBC) *TS satisfies ALBC iff for each predicate $p \in P$ and each major literal $c_i \in L_p$, TS must contain test cases making the major literal boundary in L_p be tested and c_i be evaluated to true, and c_i be evaluated to false.*

ALBC only requires cover the major literal boundary. But when the major literal evaluates to true or false, whether the minor literal boundary are required to be tested, and are evaluated to the same value. The different answers to this question have three different logical boundary coverage criteria.

Definition 9 (General Active Literal Boundary Coverage, GALBC) *TS satisfies GALBC iff for each predicate $p \in P$ and each major literal $c_i \in L_p$, TS contains at least two test cases*

making c_i boundary be tested and c_i be evaluated to true, and c_i be evaluated to false. For the minor literal boundary are not asked to be tested and are not evaluated the same value as the major literal.

For predicate p=(a≥3)∨ (b < 4), test suit {a=3∧ b=3, a=2∧ b=4} satisfies the GALBC.

Definition 10 (Correlated Active Literal Boundary Coverage, CALBC) *TS satisfies CALBC iff for each predicate p∈ P and each major literal c_i ∈ L_p, TS contains at least two test cases making c_i boundary be tested and c_i be evaluated to true, and c_i be evaluated to false. And these two test cases must cause p to be true, and p to be false.*

For the predicate p=(a≥3)∨ (b < 4), test suit {a=3∧ b=4, a=2∧ b=4, a=2∧ b=3} satisfies the CALBC. Among them a=3∧ b=4, a=2∧ b=4 test when a≥3 is the major literal; a=2∧ b=4, a=2∧ b=3 test when b < 4 is the major literal.

Definition 11 (Restricted Active Literal Boundary Coverage, RALBC) *TS satisfies RALBC iff for each predicate p∈ P and each major literal c_i ∈ L_p, TS contains at least two test cases making c_i boundary be tested and c_i be evaluated to true, and c_i be evaluated to false. The minor literal boundary is also required to be tested and evaluate to the same value as the major literal.*

Definition 12 (Literal Combinatorial Boundary Coverage, LCBC) *TS satisfy LCBC iff for each p ∈ P, TS contains test cases cause the boundary of all the literals in L_p to evaluate to all possible combination of truth values.*

The above presented logic boundary coverage criteria (as defined in 6 ~ 12) can be either one boundary or be multi-dimensional boundary. Multi-dimensional boundary is stricter than the one boundary.

Each test criteria has its advantages and disadvantages. A common way to assess test criteria is in terms of subsumption. A test criterion C_1 subsumes C_2 iff every test suit that satisfies C_1 will also satisfy C_2. Subsumption relationship is transitive. According to definition 6~12, LCBC is the most strict test criteria. It subsumes other logical boundary criteria. LBC is a weak coverage criteria, only requires at least one literal c_i boundary to be tested. PBC requires that literal c_i boundaries be tested and makes the predicate p be evaluated to true and false respectively, obviously c_i is the major literal. CALBC is similar to PBC. The difference between them is that the PBC requires at least one of the major literal boundaries be tested, but the CALBC requires each major literal boundaries be tested. According to the definition, test suit satisfies CALBC will satisfy PBC, satisfies RALBC absolutely satisfies CALBC, and satisfies CALBC will satisfy GALBC. The subsumption relation graph among these testing criteria is shown as figure 1.

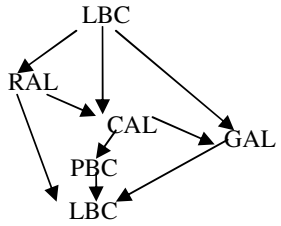

Figure 1: The subsumption relation graph among testing criteria

5 Generation of Test Cases

For the generation of test cases using logic boundary coverage criteria, we must identify the predicate and literal truth value combination which satisfy specific criteria first, then give the specific test data at the boundary state.

The selection of test points at the boundary, we refer to paper [4] which proposed the ON-OFF points selection strategies. The ON points are those input variable taking the boundary value and making the literal be evaluated to true. OFF points are those input variable taking just beyond the boundary value (boundary value ± ε) and making the literal be evaluated to false, ε≥0 is a minimum value.

Legeard et al used the industry example Smart Card GSM 11-11 standard [5], Java Card transaction mechanism [6] and Metro/RER ticket validation algorithm [7] to prove for the same number of test cases, boundary value testing can detect more errors than original testing method and can save test cost greatly. Taking advantages of logic criteria and boundary criteria, test cases based on logical boundary coverage criteria not only cover the boundaries of the system but also satisfy the logical coverage criteria. Corresponding tool prototype for test cases generation using this approach has been developed and applied to some examples. Results show test cases satisfying these criteria can detect more errors than original logic coverage criteria and reduce the number of test

cases. For space limitation, this paper does not illustrate and give the empirical evaluation.

6 Conclusions and future work

Boundary testing is one of most effective functional testing strategies. This paper combines the boundary coverage criteria and the logic coverage criteria, proposes a series of model-based logic boundary coverage criteria. Test cases based on these criteria can satisfy corresponding logic coverage criteria and detect more errors. Using these criteria to generate test cases also can reduce the number of test cases and improve test efficiency. Our future work includes improving the proposed criteria and the automatic prototype tool.

Acknowledgements

This work is supported by National Natural Science Foundation of China (NSFC) under grant No. 60970007, the National Grand Basic Research Program (973 Program) of China under grant No. 2007CB310800, the Natural Science Foundation of Shanghai Municipality of China under Grant No.09ZR1412100, Science and Technology Commission of Shanghai Municipality under Grant No. 10510704900 and the Shanghai Leading Academic Discipline Project of China under Grant No.J50103.

References

[1] N. Kosmatov, B. Legeard, F. Peureux and M. Utting, Boundary coverage criteria for test generation from formal models. In Proc. of the 15th Int'l Symposium on Software Reliability Engineering. (2004), 139–150.

[2] Amman P and Offutt J, Coverage criteria for logical expressions. In Proc. of the 14th Int'l Symposium on Software Reliability Engineering. Washington: IEEE Computer Society Press. (2003), 99-107.

[3] Liu L and Miao HK, Axiomatic assessment of logic coverage software testing criteria. Journal of Software. (2004), 15(9), 1301-1310.

[4] B.Jeng, Elaine J.Weyuker, A simplified domain testing strategy, ACM Transactions on Sotfware Engineering and Methodology, Vol.3, No.3. (1994), 254-270.

[5] B. Legeard, F. Peureux, and M. Utting, Automated Boundary Testing from Z and B. In Proc. of the International Conference on Formal Methods Europe (FME'02), volume 2391 of LNCS, Copenhagen, Denmark, Springer Verlag. (2002), 21-40.

[6] F. Bouquet and B. Legeard, Reification of executable test scripts in formal specification-based test generation: the Java Card transaction mechanism case study. In Proceedings of the International Conference on Formal Methods Europe (FME'03), volume 2805 of LNCS, Pisa, Italy, Springer Verlag. (2003), 778-795.

[7] N. Caritey, L. Gaspari, B. Legeard, and F. Peureux, Specification-based testing- Application on algorithms of Metro and RER tickets (confidential). Technical Report TR-03/01, LIFC-University of Franche-Comt´e and Schlumberger Besanc, on, (2001).

Liping Li received the Master degree in software engineering from Tongji University, Shanghai, China, in 2006. Now study the PhD of computer application technology in Shanghai University from 2007. She is currently an associate professor in the Shanghai Second Polytechnic University. Her research interests include software testing and software engineering.

Huikou Miao received the Master degree in computer Application Technology from Shanghai University of Science and Technology, Shanghai, China, in 1986. He is currently a professor in computer Engineering and Science at Shanghai University, China. His research interests include software formal methods and software engineering.