

Identifying Categories and Choices for Software Testing Based on Informal Specifications

Sau Fun TANG

Submitted in total fulfilment of the requirements
of the degree of Doctor of Philosophy

March 2009

Faculty of Information and Communication Technologies
Swinburne University of Technology

Abstract

It would be ideal to test a program with all its possible inputs (that is, the *input domain*). In this way, any fault that may exist in the program would be guaranteed to be detected. In practice, however, this “exhaustive” approach is difficult, or almost impossible, to apply because of limited testing resources. Therefore, a more feasible approach is to construct a *test suite*, which is a subset of the input domain, for testing. The fundamental problem of this approach is how to construct a test suite so that the comprehensiveness of the test cases, as compared to the input domain, is jeopardized as less as possible.

To solve this problem, two closely related test suite generation techniques have been developed and have attracted much attention. They are the **CHOiCe reLATion framEwork** (abbreviated as CHOC’LATE) and the **Classification-Tree Methodology** (abbreviated as CTM). A main advantage of both techniques is that they can be applied to informal specifications, which are commonly used in the commercial software industry.

In CHOC’LATE and CTM, an early step is to identify a set of categories and choices from the specification, through which a test suite is generated. This set of categories and choices is very important because it affects the comprehensiveness of the test suite, which in turn affects the chance of detecting software faults. We observe that neither the original developers of CHOC’LATE/CTM nor follow-up researchers have proposed a systematic method for identifying categories and choices from informal specifications. As a result,

the identification process is often performed in an ad hoc, impromptu manner. Without doubt, an impromptu approach cannot assure the quality of the identified categories and choices. This thesis addresses such a problem, with a view to developing an effective methodology for identifying categories and choices from informal specifications.

To provide the background motivation, we conduct two rounds of empirical studies using several commercial specifications primarily written in an informal manner. The first round of studies investigates the common mistakes made by *inexperienced* software testers in an impromptu identification approach. This round of studies serves three purposes: (a) to make the knowledge of common mistakes known to other testers so that they can avoid repeating the same mistakes, (b) to facilitate researchers and practitioners develop systematic identification methodologies, and (c) to provide a means of measuring the effectiveness of newly developed identification methodologies. We also present a checklist to help testers detect the common mistakes. In the second round of studies, we shift our focus to *experienced* testers with many years of commercial experience in software development and testing. The objectives of the studies are: (a) to investigate the differences in the types and amounts of mistakes made between inexperienced and experienced testers in an impromptu identification approach, and (b) to determine, after discussing the mistakes with the testers and providing them with a checklist as a simple guideline for detecting problematic categories and choices, the extent of mistake reduction in the next identification exercise. We also discuss which specification components are useful for category and choice identification.

Thereafter, we present the most significant work of the thesis: our **DividE-and-conquer** methodology for identifying categorie**S**, choice**S**, and choic**E** **R**elations for **T**est case generation (abbreviated as **DESSERT**). The methodology is particularly useful for informal specifications which are complex and contain many different types of specification components. The theoretical framework and the associated algorithms of **DESSERT** will

be discussed in detail. We shall also describe two case studies using commercial specifications to demonstrate the effectiveness of DESSERT.

Acknowledgement

Firstly, I wish to express my gratitude to my coordinating supervisor, Prof. T. Y. Chen, and associate supervisors, Prof. T. H. Tse and Dr. F.-C. Kuo, who helped me throughout this work by providing constant guidance, support, and stimulating feedback.

Acknowledgement is due to the Department of Computer Science of The University of Hong Kong, where part of my research was performed. I am also indebted to Asso. Prof. Y. T. Yu in the Department of Computer Science at City University of Hong Kong and Asso. Prof. P. L. Poon in the School of Accounting and Finance at The Hong Kong Polytechnic University for their invaluable help and insightful advice. I also thank the Faculty of Information and Communication Technologies at Swinburne University of Technology and their staff for the assistance and support offered to me.

I gratefully acknowledge the financial support from a Swinburne University Postgraduate Research Award during part of my Ph.D. candidature. Part of the work in this thesis was supported by a research grant (project no. 123206) from the Hong Kong Research Grants Council.

Declaration

I hereby declare that this thesis comprises the original work undertaken during my Ph.D. candidature. The thesis contains no material which has been accepted for the award to me of any other degree or diploma. In addition, to the best of my knowledge, the thesis contains no material previously published or written by another person except where due reference is made in the text of the thesis. Parts of this thesis are the results of joint research or collaboration with others [10, 11, 15, 16, 18, 49, 62].

I also certify that the thesis is less than 100 000 words in length, excluding tables, footnotes, and bibliographies.

Signature: _____

Sau Fun Tang

Contents

List of Tables	ix
List of Figures	xii
1 Introduction	1
1.1 Importance of Software Testing	1
1.2 Two Approaches to Test Suite Generation	3
1.2.1 White-Box Testing	3
1.2.2 Black-Box Testing	6
1.3 Motivation of Thesis	10
1.4 Overview of Thesis	11
2 Literature Review	14
2.1 Overview of Choice Relation Framework (CHOC'LATE)	14
2.2 Overview of Classification-Tree Methodology (CTM)	22
2.3 Major Problems	25
3 Our Two Rounds of Empirical Studies	28
3.1 First Round of Studies	28
3.1.1 Purposes of Studies	28
3.1.2 Overview of Function Models and Function Rules	29

3.1.3	Terminology and Definitions	30
3.1.4	Experimental Setting	44
3.1.5	Findings, Discussions, and Recommendations	46
3.1.6	Threads to Validity	51
3.1.7	Summary	52
3.2	Second Round of Studies	53
3.2.1	Purposes of Studies	53
3.2.2	Experimental Setting	54
3.2.3	Study 1: Effectiveness of Tester Experience	55
3.2.4	Study 2: Effectiveness of Checklist Guideline	70
3.2.5	Feedback of Subjects	73
3.2.6	Threads to Validity	81
3.2.7	Summary	82
4	Our Identification Methodology: DESSERT	84
4.1	Overall Approach of DESSERT	84
4.2	Construction of Preliminary Choice Relation Tables	85
4.2.1	Overview of Activity Diagrams	86
4.2.2	Table Construction from Activity Diagrams	87
4.3	Consolidation of Preliminary Choice Relation Tables	92
4.3.1	Step (3.1) of DESSERT	93
4.3.2	Step (3.2) of DESSERT	101
4.4	Case Study 1	140
4.4.1	Setting of Study	140
4.4.2	Constructing a Preliminary Choice Relation Table from an Activ- ity Diagram	141

4.4.3	Constructing Preliminary Choice Relation Tables from Other Specification Components	143
4.4.4	Consolidating Preliminary Choice Relation Tables	145
4.5	Case Study 2	148
4.5.1	Setting of Study	148
4.5.2	Constructing Preliminary Choice Relation Tables from Activity Diagrams	150
4.5.3	Constructing Preliminary Choice Relation Table from Other Specification Components	150
4.5.4	Consolidating Preliminary Choice Relation Tables	152
4.6	Summary	154
5	Conclusions	156
5.1	Summary of Results and Contributions	156
5.2	Future work	158
	Bibliography	160

List of Tables

2.1	Categories and Choices for $\mathbb{U}_{\text{COUNT}}$	18
3.1	A Partial Function Model for Program COUNT	30
3.2	Statistics of Potential Categories and Potential Choices Identified for Each Functional Unit	47
3.3	Statistics of Missing and Problematic Categories for Each Functional Unit	48
3.4	Numbers and Percentages of Different Types of Problematic Category . .	49
3.5	Numbers and Percentages of <i>PC</i> 's Containing Different Types of Problematic Category	49
3.6	Statistics of Potential Categories and Choices Identified by Inexperienced and Experienced Testers	57
3.7	Total Numbers, Mean Numbers, and Mean Percentages of Missing Categories by Inexperienced and Experienced Testers	59
3.8	Percentage Increase in Mean Numbers of Missing Categories by Inexperienced and Experienced Testers	60
3.9	Percentage Increase in Mean Numbers of Potential Categories Identified by Inexperienced and Experienced Testers	61
3.10	Total Numbers, Mean Numbers, and Mean Percentages of Problematic and Non-Problematic Categories Identified by Inexperienced and Experienced Testers	62

3.11	Percentage Increase in Mean Numbers/Percentages of Problematic Categories Identified by Inexperienced and Experienced Testers	64
3.12	Percentage Increase/Decrease in Mean Numbers/Percentages of Non-Problematic Categories Identified by Inexperienced and Experienced Testers	66
3.13	Total Numbers of Different Types of Problematic Category Identified by Inexperienced and Experienced Testers	67
3.14	Mean Numbers of Different Types of Problematic Category in Each <i>PC</i> Identified by Inexperienced and Experienced Testers	67
3.15	Percentages of Different Types of Problematic Category in All Potential Categories Identified by Inexperienced and Experienced Testers	68
3.16	Percentages of Different Types of Problematic Category in All Problematic Categories Identified by Inexperienced and Experienced Testers	68
3.17	Total Numbers of Missing Categories with and without Checklist	71
3.18	Total Numbers of Different Types of Problematic Category with and without Checklist	72
4.1	Preliminary Choice Relation Table τ_1	102
4.2	Preliminary Choice Relation Table τ_2	102
4.3	Choice Relations Stored in Associated Linked Lists in Figure 4.1(a)	104
4.4	Choice Relations Stored in Associated Linked Lists in Figure 4.1(b)	105
4.5	Possible Combinations of Two Choice Relations Involving Overlapping Choices	118
4.6	Interim Choice Relation Table \mathcal{T}_1 Constructed after Step (3)(b) of re-fine_table	139
4.7	Number of Potential/Missing/Problematic Categories and Choices for \mathbb{S}_{VAS} (Before Correction)	142

4.8	Number of Relevant Categories and Valid Choices for \mathbb{S}_{VAS} (After Correction)	144
4.9	Number of Relevant Categories and Valid Choices for \mathbb{S}_{CODE} (After Correction)	152

List of Figures

1.1	Spectrum of Different Forms of Specifications	7
2.1	Classification Tree Υ_{COUNT}	24
3.1	The Function Model of a Software System	29
3.2	Part of an Activity Diagram for the Generation of Daily Meal Schedules .	75
3.3	A Level-1 Data Flow Diagram for the Generation of Daily Meal Schedules	78
4.1	Structures of L and its Associated LL_{m_i} 's During and After the Execution of integrate_table	103
4.2	Some Possible Relations among $TC(X : x_1)$, $TC(X : x_2)$, and $TC(Y : y)$ When $ X : x_1 \cap X : x_2 \neq \emptyset$, $ X : x_1 \not\subseteq X : x_2 $, $ X : x_2 \not\subseteq X : x_1 $, and $ X : x_1 \sqsupseteq Y : y $	110
4.3	Some Possible Relations among $TC(X : x_1)$, $TC(X : x_2)$, and $TC(Y : y)$ When $ X : x_1 \subsetneq X : x_2 $ and $ X : x_2 \sqsupseteq Y : y $	111
4.4	Some Possible Relations among $TC(X : x)$, $TC(Y : y_1)$, and $TC(Y : y_2)$ When $ Y : y_1 \cap Y : y_2 \neq \emptyset$, $ Y : y_1 \not\subseteq Y : y_2 $, $ Y : y_2 \not\subseteq Y : y_1 $, and $ X : x \sqsubset Y : y_1 $	114
4.5	Some Possible Relations among $TC(X : x)$, $TC(Y : y_1)$, and $TC(Y : y_2)$ When $ Y : y_1 \cap Y : y_2 \neq \emptyset$, $ Y : y_1 \not\subseteq Y : y_2 $, $ Y : y_2 \not\subseteq Y : y_1 $, and $ X : x \sqsupseteq Y : y_1 $	114

4.6	Some Possible Relations among $TC(X : x)$, $TC(Y : y_1)$, and $TC(Y : y_2)$ When $ Y : y_1 \subsetneq Y : y_2 $ and $ X : x \sqsubset Y : y_2 $	116
4.7	Some Possible Relations among $TC(X : x)$, $TC(Y : y_1)$, and $TC(Y : y_2)$ When $ Y : y_1 \subsetneq Y : y_2 $ and $ X : x \sqsupseteq Y : y_2 $	117
4.8	A Partial Activity Diagram $\mathcal{A}_{\text{REQUEST}}$	146