T-WAY TESTING : A TEST CASE GENERATOR BASED ON MELODY SEARCH ALGORITHM (TTT-MS)

TOH SHU YUEN

BACHELOR OF COMPUTER SCIENCE

(SOFTWARE ENGINEERING)

UNIVERSITI MALAYSIA PAHANG

ABSTRACT

Nowadays, the era of advanced technology has brought modern software systems into our daily life to satisfy the various needs of users. To ensure a well function of software systems without defects, software testing plays a dominant role on it. However, exhaustive testing is impossible that is required a huge amount of cost and time. This thesis is research on developing a T-Way Testing : A Test Case Generator Based On Melody Search Algorithm (TTT-MS) to generate optimum final test list. First, Inputs will be written into a text file and TTT-MS will read the inputs which are the number of Parameters, number of values for each Parameter, Parameter Names, Parameter Values Name and the Interaction Strength (t) up to t=4. Next, TTT-MS will be executed through main algorithms to generate Parameters Interaction List, Parameter Values Interaction List, and finally generate final Test Cases based on Melody Search Algorithm. Basically, TTT-MS consists of four main procedures which are User Interface, Interaction List Generation, Test Case Generation and Mapping of Parameter Values in Symbolic Value to Actual Name. After the generation of Final test List through the execution of TTT-MS, the results are being evaluated and compared with other existing t-way strategies. Results shows that sometimes TTT-MS performed better than other existing strategies in certain data specification due to the behavior of optimization algorithms.

ABSTRAK

Pada masa kini, era teknologi yang canggih telah membawa sistem perisian moden ke dalam hidup kita untuk memenuhi pelbagai keperluan harian pengguna. Untuk memastikan system perisian berfungsi dengan lancar tanpa kecacatan, ujian perisian memainkan peranan dominan. Walau bagaimanapun, ujian perisian dengan sepenuhnya yang merangkumi setiap interaksi nilai parameter satu demi satu adalah mustahil memandangkan sejumlah besar wang dan masa diperlukan. Tesis ini akan membincang tentang uji kaji untuk mencipta T-Way Testing : A Test Case Generator Based On Melody Search Algorithm (TTT-MS) bagi menghasilkan koleksi set pengujian yang optimum. Pertama, input sebagai bilangan parameter, bilangan nilai untuk setiap parameter, name untuk setiap parameter, name untuk nilai setiap parameter dan nilai t sampai t=4. TTT-MS akan dilaksanakan melalui beberapa algoritma penting untuk menghasilkan senarai interaksi parameter, senarai interaksi nilai parameter, dan akhirnya set pengujian akhir dengan Melody Search Algorithm. TTT-MS terdiri daripada empat procedure utama iaitu User interface, Senarai interaksi parameter, Senarai set pengujian akhir, dan Pemetaan Nilai Parameter dalam Simbolik kepada Nama Sebenarnya.Setelah set pengujian akhir dikeluarkan, hasil ini digunakan untuk dinilai dan dibandingkan dengan strategi lain yang sedia ada, prestasi TTT-MS terbukti mampu menghasilkan koleksi set pengujian yang optimum dengan mengurangkan bilangan set pengujian. Kadankala TTT-MS manghasilkan hasil yang lebih baik daripada strategi lain dengan faktor kelakuan algoritma pengoptimuman.

TABLE OF CONTENTS

CHAPTER 3 METHODOLOGY

CHAPTER 4 DESIGN, IMPLEMENTANTION AND RESULT DISCUSSION

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACA	Ant Colony Algorithm
ACS	Ant Colony System
AETG	Automatic Efficient Test Generator
AI	Artificial Intelligence
BA-PTCS	A Bat-Inspired Strategy for Pairwise Testing With Constraints Support
Bw	Constant distance bandwidth
CA	Covering Array
CCA	Constraints Covering Array
CP	Combinatorial Optimization Problem
CT	Combinatorial testing
DDA	Deterministic Density Algorithm
GA	Genetic Algorithm
HM	Harmony Memory
HS	Harmony Search Algorithm
IEEE	Institute of Electrical and Electronics Engineers
IPO	In-Parameter-Order
IPRS	Intersection Residual Pair Set
LAHC	Late Acceptance Hill Climbing Algorithm
MCA	Mixed Covering Array

MCCA	Mixed Constraints Covering Array
MM	Melody Memory
MS	Melody Search Algorithm
NII	Maximum number of iterations for initial/first phase
PAR	Pitch Adjusting Rate
PICT	Pairwise Independent Combinatorial Testing
PM	Player Memory
PMCR	Player Memory Considering Rate
PMN	Number of player memories
PMS	Player memory size
RAD	Rapid Application Development
SA	Simulated Annealing Algorithm
SDLC	Software Development Life Cycle
SUT	System Under Test
TVG	Test Vector Generator
VCA	Variable-strength Covering Array

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Nowadays, the era of advanced technology has brought modern software systems into our daily life to satisfy the various needs of users. To ensure a well function of software systems without defects which can leads to failure and malfunction, software testing plays a dominant role on it. It has been known for some time that, in a typical programming project, approximately 50% of elapsed time and over 50% of total cost are expended in testing part (Dalley, 1991).

According to definition of Institute of Electrical and Electronics Engineers (IEEE), Software Testing is a process to analyze a software system to determine the differences between the requirements and the existing condition of the software like defects or bugs and evaluate the defects found in the software system. In other words, the main objective of software testing is to perform the verification of software system so that it is well-developed according to the requirements and free from failure caused by defects.

Combinatorial testing (CT) is one of the testing strategy which selecting test case by combining the values of different input parameters using some combinatorial algorithm strategy (Mats Grindal, 2004). However, it is often impractical to execute exhaustively by testing all the combinations due to complexity, time-consuming, resource constraints and require a huge amount of cost as well (Shiwei Gao, 2014). The size of a test list required to test all of the possible combinations can be impossible in even a moderate size of project

(Myra B. Cohen, 2003). Furthermore, to generate a combinatorial test list which consists of more than 6-ways combination is very expensive also (Kiran Shakya, 2012).

Therefore, it is crucial to propose another strategy to reduce the number of test cases in order to reduce the cost and time needed in software testing. With this, t-way combinatorial testing which generate test case covered all t combinations of values of every parameters of a System Under Test (SUT) is encouraged (Shiwei Gao, 2014). This testing strategy generated smaller number of test list based on t-way combinations of input values of t parameters where t also represents the strength of interaction. Each t-way combination of valid values of these t parameters is covered by at least one test case (R. K. Yu Lei, D.Richard Kuhn, Vadim Okun, James Lawrence, n.d.).

There are many existing t-way strategies which includes pure computational-based like Test Vector Generator (TVG) (P.J. Schroeder, 2003) , ParaOrder (Z.Y. Wang, 2008) and Artificial Intelligence (AI) – based like Genetic Algorithm (McCaffrey, 2010; R. Bryce, 2007; Sthamer, 1995; Toshiaki Shiba, 2004; W. Afzal, 2009) and Harmony Search Algorithm (Abdul Rahman A. Alsewari, 2012). With this, this research proposed t-way Testing : A Test Case Generator based on another algorithm under Artificial Intelligence (AI) based called Melody Search algorithm which will be further discussed in details in Chapter 4.

1.2 PROBLEM STATEMENT

During testing, we need to optimize or reduce the test cases so that we do not repeat certain interaction of values. This problem is considered as Combinatorial Optimization (CP) Problem. Combinatorial optimization problem involves the process of finding optimal solutions or minimum number of test suite.

For example, TSY food ordering system consists of 5 parameters with 2 three-valued parameters and 3 two-valued parameters as shown in Table 1. To test all the possible combinations exhaustively, the number of test case will be 3^2 x 2^3 (v^p) where v refers to the number of values and P refers to the number of parameters, so 3^2 x 2^3 = 72 test case. In other words, if the system continues to expand to become 10 parameters with 7 three-valued parameters and 3 two-valued parameters, the number of test case will become 3^7 x 2^3 = 17496 test cases. It will become impossible as it require a huge amount of cost and time to test.

Table 1 Components of TSY Food Ordering System

Food	**Flavor**	**Add Ingredients**	**Delivery**	**Payment**
Mee	Spicy	Egg	Eat in	Cash
Fried Rice	Not Spicy	Chicken	Take Away	Credit Card
Meehun		None		

Figure 1 is another example to shows how optimization algorithms reduce the number of test cases by creating optimum test list. It is an order option of food ordering system consisting of 4 parameters and 2 values for each parameter. Each parameter and value is being assigned with an unknown term as Table 2 shown.

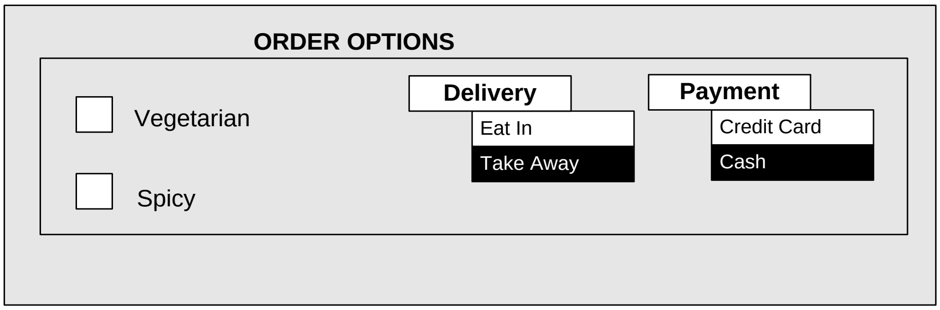

Figure 1 An order options of food ordering system

Table 2 Parameters and Values of food ordering system

Vegetarian, P1	Spicy, P2	Delivery, P3	Payment, P4
Checked, a1	Checked, b1	Eat in, c1	Credit Card, d1
Unchecked, a2	Unchecked, b2	Take Away, c2	Cash, d2

Exhaustive testing with full interaction strength is 2 X 2 X 2 X 2 = 16. This means that 16 test cases will be created as shown in Table 3.

Table 3 Test list of Exhaustive Testing

Test Case	P1	P2	P3	P4
1	a1	b1	c1	d1
2	a1	b1	c1	d2
3	a1	b1	c2	d1
4	a1	b1	c2	d2
5	a1	b2	c1	d1
6	a1	b2	c1	d2
7	a1	b2	c2	d1
8	a1	b2	c2	d2
9	a2	b1	c1	d1
10	a2	b1	c1	d2
11	a2	b1	c2	d1
12	a2	b1	c2	d2
13	a2	b2	c1	d1
14	a2	b2	c1	d2
15	a2	b2	c2	d1
16	a2	b2	c2	d2

If we consider the interaction strength as t=3 by random selection, the possible interaction of parameters are {P1, P2, P3}, {P1, P2, P4}, {P1, P3, P4} and {P2, P3, P4}. The possible test list create are shown in Table 4, Table 5, Table 6 and Table 7.

Table 4 Test list of {P1, P2, P3} parameters interaction

Test Case	P1	P2	P3
1	a1	b1	c1
2	a1	b1	c2
3	a1	b2	c1
4	a1	b2	c2
5	a2	b1	c1
6	a2	b1	c2
7	a2	b2	c1
8	a2	b2	c2

Table 5 Test list of {P1, P2, P4} parameters interaction

Test Case	P1	P2	P4
1	a1	b1	d1
2	a1	b1	d2
3	a1	b2	d1
4	a1	b2	d2
5	a2	b1	d1
6	a2	b1	d2
7	a2	b2	d1
8	a2	b2	d2

Table 6 Test list of {P1, P3, P4} parameters interaction

Test Case	P1	P3	P4
1	a1	c1	d1
2	a1	c1	d2
3	a1	c2	d1
4	a1	c2	d2
5	a2	c1	d1
6	a2	c1	d2
7	a2	c2	d1
8	a2	c2	d2

Table 7 Test list of {P2, P3, P4} parameters interaction

Test Case	P2	P3	P4
1	b1	c1	d1
2	b1	c1	d2
3	b1	c2	d1
4	b1	c2	d2
5	b2	c1	d1
6	b2	c1	d2
7	b2	c2	d1
8	b2	c2	d2

Lastly, optimization can be carry out by consider similar interaction of parameters classified under one new test case until an optimum test list which covered all the possible interaction between values is created as shown in Table 8. The total number of test cases has reduced from 16 to 8 test cases.

Table 8 Optimum Test List with interaction strength t=3

Test case	P1	P2	P3	P4	Weight
1	a1	b1	c1	d1	4
2	a1	b1	c2	d2	4
3	a1	b2	c1	d1	4
4	a1	b2	c2	d2	4
5	a2	b1	c1	d1	4
6	a2	b1	c2	d2	4
7	a2	b2	c1	d1	4
8	a2	b2	c2	d2	4

So, in order to reduce and optimize the test case, this research is suggesting another optimization algorithm called Melody Search Algorithm to be used as the engine search technique for the proposed combinatorial test data generation strategy with supporting interaction strength (t) up to t=4 in this project.

1.3 OBJECTIVES

Several objectives have been set as following so to achieve my goal of developing a T-way Testing Generator Based on Melody Search Algorithm (MS) :

- To investigate the adoption of Melody Search Algorithm and apply in test list generation.
- To support higher range of interaction strength (t) up to t=4 with customize values of each parameter.
- To evaluate TTT-MS with existing strategies based on the test list size.

1.4 SCOPE

The scopes of research on TTT-MS are as following :

- t-way testing test case generator based on Melody Search Algorithm.
- t-way testing test case generator with range of interaction strength (t) up to t=4 with customize values of each parameter.
- Develop t-way testing test case generator using Java programming language.

1.5 THESIS ORGANIZATION

This thesis is consists of seven main chapters with its own discuss topics. Chapter 1 is Introduction. In Chapter 1, topic of this research will be introduced roughly. Introduction, problem statement, objectives, scope and thesis organization will be known in this chapter.

Chapter 2 is Literature Review. Research on the existing t-way testing generating tool algorithm will be discussed on this chapter.

Chapter 3 is Methodology. Software Development Methodology is being determined on this chapter. Its development process will be further discussed on this chapter.

Chapter 4 is Implementation, Testing and Result Discussion. The concept of framework, design approach, implementations and result discussion of TTT-MS is discussed in this chapter. The development processes of TTT-MS in flowchart and algorithm will also be present in this chapter. Besides, the result generated from TTT-MS will also be evaluated and compare with other existing t-way strategies discussed in literature review.

Chapter 5 is Conclusion. This chapter will summarize the research on its constraints and future work to propose any suggestion for future improvement on this research.

1.6 SUMMARY

In short, this chapter has discussed about the title of this research and introduced about one of the t-way strategy used in this research which is Melody Search (MS) Algorithm to generate an optimized test list based on t-way combinations of input values of t parameters where t also represents the strength of interaction. The problem statement, objectives, scope and thesis organization has discussed and defined in this chapter also.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

In Chapter 1, we has discussed about the introduction of this research which included the problem statement, Objectives, Scope and Thesis Organization. With the information in Chapter 1, Chapter 2 is going to focus on Covering Array Notification and the existing literature review.

2.2 COVERING ARRAY NOTATIONS

Although exhaustive testing may be possible for small number of parameters and values for each parameter, yet a wise selection of test case to minimize the size of a test list is needed for large number of parameters and values. Covering arrays are being used to deal with the combinatorial testing.

2.2.1 Covering Array (CA)

Covering Array is consists of four elements which are N, t, P, v ; CA (N, t, v^P) where N represent the size of test case; t refers to the interaction strength; P refers to the number of parameters and v refers to the number of values of each parameter. (Abdul Rahman A. Alsewari, 2012) Normally, a t-way interaction test list is a covering array (Jianjun Yuan, 2010).

For example, CA (8, 2, 3^5) refers to a test list consists of 8 x 5 arrays where the row = 8 represents the size of the test case (N) and column = 5 represents the number of parameter (P). This test list consists of five 3-values parameters undergo two-way (t=2) interactions.

Covering array only suitable for the needs of tests where all components of software have the same number of configurations which means all parameters have the same number of values (Myra B. Cohen, 2003).

2.2.2 Mixed Covering Array (MCA)

Mixed Covering Array has three elements which are N, t, C ; MCA (N, t, C) where N represent the size of test case; t refers to the interaction strength and C refers to the configuration. Similar to the earlier notations of CA, Configuration (C) represent the parameter and value of each configuration presented in the format of (V_1^{P1}, V_2^{P2}, V_3^{P3},…… V_n^{Pn}) where it indicates that P_1 parameter has v_1 values, P_2 parameter has v_2 values, P_3 parameter has v_3 values and so on (Abdul Rahman A. Alsewari, 2012).

For example, MCA (128, 3, 8^2 3^2 5^4 7^1) refers to a test list consists of test size of 128 with the configuration of 9 parameters (2 eight-valued parameters, 2 three-valued parameters, 4 five-valued parameters and 1 seven-valued parameter) undergo three-way interaction (t=3).