DEVELOPMENT OF EFFICIENT T-WAY TEST DATA GENERATION ALGORITHM AND EXECUTION STRATEGIES

KHANDAKAR FAZLEY RABBI

Thesis submitted in fulfillment of the requirements
For the award of the degree of
Master of Computer Science

UNIVERSITI MALAYSIA PAHANG

SEPTEMBER 2012

ACKNOWLEDGEMENTS

The work presented in this thesis was undertaken under the main supervision of Prof. Dr. Sabira Khatun, to whom I am grateful for her support, interest and her insightful and critical comments in all stages of the work. To say the least, without her encouragement and enthusiasm, this work may not be done with this strength. Also, even she is very busy; she took an enormous task of revising my thesis word by word. Her efforts are greatly appreciated and will never be forgotten. Thanks again Prof.

I am also grateful to En. Che Yahaya for his help and co-operation during this work.

I am also thankful to all of my friends in Malaysia, who gave their support and help through many helpful and enjoyable discussions. In particular, I am thankful to all academic staffs in the Faculty of Computer Science and Software Engineering, UMP, and all those persons who have encouraged me to complete my study. Thanks!

I am also grateful to my parents, brothers and sisters, uncles and aunts for strong encouragement in my studies, so that I could be successful in my life. Dad and mom, thank you for the prayers – this thesis is for both of you.

Finally, I would like to thank my loving wife (Rupu). To my daughter (Rijja), thanks for being patient all along. I am sorry to have sometimes neglected all of you to pursue my dream.

ABSTRACT

For a typical software product, it is desired to test all possible combinations of input data in various configurations, Exhaustive testing is impossible to execute prior to release in the market. The lack of resources, cost factors and tight deadlines to market are some of the main factors that prevent this consideration. In current practice, usually the test-data are selected and executed randomly. Many useful strategies (2-Way and T-Way sampling) were developed to generate test-data and facilitate smooth testing process. Comprehensive test data generation is nondeterministic polynomial hard problem (NP-complete). Hence, optimization in terms of number of generated test-data and execution time is in demand. Motivated by these, this thesis addresses the design, implementation and validation of four effective test data generation strategies in terms of 2-way and T-way as follows: (I) PS2Way (Pairwise Search for 2-way Test Data Generation) strategy is based on parameters pair technique. This strategy is able to reduce the number of test data, but the execution time is not optimized. (II) EasyA (Easy Algorithm for 2-way Test Data Generation) is developed based on matrix based calculation to overcome the time constrains by keeping the number of test data in an acceptable range. This strategy is unable to support non uniform values. (III) R2Way (Random Search for 2-way Test Data Generation) with execution tool is developed to support both uniform and non-uniform values using search based software engineering. R2Way outperforms other existing strategies but cannot support higher interaction level ($T > 2$). And finally, (IV) MTTG (Multi-Tuple based T-way Test Data Generation) strategy is developed inspiring kids “Card Game” to overcome the limitations of interaction strength. An executable prototype tool is also developed for auto test execution besides efficient test data generation. Empirical data shows that MTTG is effective and outperforms other strategies in terms of number of test data generation time (more than 74% improvement), maintaining a tolerable test data size. All the proposed strategies are simpler to implement and handle. They are also efficient in terms of execution time and able to generate highly reduced test data suites to fulfil the current demand (easy and faster process) by software development companies.

ABSTRAK

Bagi setiap produk perisian yang biasa, ia hendaklah menguji semua kemungkinan kombinasi data input dalam pelbagai konfigurasi, iaitu ujian menyeluruh sebelum ia dipasarkan. Kekurangan sumber, faktor-faktor kos dan tarikh akhir yang ketat untuk ke pasaran adalah antara faktor utama yang menghalang pertimbangan ini. Dalam amalan semasa, biasanya data ujian dipilih dan dilaksanakan secara rawak. Pelbagai strategi yang berguna (2-hala dan pensampelan T-hala) telah dibangunkan untuk menjana data ujian dan memudahkan proses ujian yang lancar. Penjanaan data ujian yang komprehensif adalah masalah nondeterministic polinomial (NP-lengkap) yang susah. Oleh itu, pengoptimuman dan pelaksanaan dari segi bilangan masa data ujian adalah permintaan yang terdesak. Didorong oleh masalah ini, tesis ini telah menunjukkan rekabentuk, pelaksanaan dan pengesahan empat strategi ujian generasi data dari segi 2-hala dan T-hala yang berkesan seperti berikut: (I) Strategi PS2Way (Pencarian berpasangan untuk Penjanaan Data Ujian 2-hala) berdasarkan teknik sepasang parameter. Strategi ini dapat mengurangkan bilangan data ujian, tetapi masa pelaksanaan tidak dioptimumkan. (II) EasyA (Algoritma Mudah untuk Generasi Data Ujian 2-hala) dibangunkan berdasarkan pengiraan berasaskan matriks untuk mengatasi kekangan masa dengan menjaga bilangan data ujian dalam julat yang boleh diterima. Strategi ini tidak dapat untuk menyokong nilai-nilai bukan seragam. (III) R2Way (Pencarian Rawak untuk Penjanaan Data Ujian 2-hala) dengan perisian dibangunkan untuk menyokong kedua-dua nilai yang seragam dan tidak seragam dengan menggunakan carian berdasarkan kejuruteraan perisian. R2Way melebihi prestasi strategi lain yang sedia ada tetapi tidak boleh menyokong interaksi tahap yang lebih tinggi (T> 2). Dan akhirnya, (IV) strategi MTTG (Multi-tuple berdasarkan Data Generasi Ujian T-Way) di inspirasikan dari "Permainan Kad" kanak-kanak untuk mengatasi keterbatasan kekuatan interaksi. Satu alat prototaip boleh laku juga dibangunkan untuk pelaksanaan ujian auto selain generasi data ujian yang cekap. Data empirikal menunjukkan bahawa MTTG adalah berkesan dan melebihi prestasi strategi yang lain dari segi generasi bilangan ujian masa data (lebih daripada 74% peningkatan), mengekalkan saiz data ujian yang diterima. Semua strategi yang dicadangkan adalah mudah untuk dilaksanakan dan dikendalikan. Ianya juga cekap dari segi masa pelaksanaan dan mampu mengurangkan penjanaan ujian sederetan data untuk memenuhi permintaan semasa (proses yang mudah dan lebih cepat) oleh syarikat-syarikat pembangunan perisian.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

V & V	Verification and Validation
NP-hard	Non-deterministic polynomial-time hard
PS2Way	Pair Parameter Search based 2 way strategy
R2Way	Random search based 2 way strategy
EasyA	Easy Algorithm
MTTG	Multi Tuple based T way Generation
IPO	In Parameter Order
TCG	Test Case Generator
mTCG	Modified Test Case Generator
TConfig	T way test configuration
CTS	Combinatorial Test Services
AETG	Automatic Efficient Test Generator
mAETG	Modified Automatic Efficient Test Generator
GA	Genetic Algorithms
ACA	Ant Colony Algorithm
IPOG	Generation of In Parameter Order
TVG	Test Vector Generator
ITCH	Intelligent Test Case Handler
GTWAY/G2WAY	Generation Of Test Case

SBSE	Search Based Software Engineering
PSO	Particle Swarm Intelligence
NA	Not Available
NS	Not Supported
TCAS	The common Traffic Collision Avoidance System

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Computing technology has highly improved these days. From type writing to telephony was once manual have been automated nowadays. Although mobile phones, washing machines, robots are hardware but it controls through a software system. The software now also became automated and intelligent. And all above the software and hardware is now became an integral part of our life.

Can we imagine our lives without software? Our favourite facebook, iphone, ipad, windows/linux/Macbook is our life partner. We can’t think of our life without those. Our favourite iphone without software will be too limited to use for our life. Thus we can divide our digital life into two digital items. One is the software and another is the hardware.

The software itself cannot do anything without hardware and software can’t run by itself. A software is always be driven by any hardware. For any hardware there have some fixed maintenance costs. But for software maintenance comes with lot of things from the first line of coding where software gets its life. The software itself can also customize whenever needed. So even though programmer writes bad/bug codes it can be fixed anytime. Although one can’t produce error free software but through a proper testing the quality (i.e. software is reliable enough to meet the requirement) and maintenance cost can be highly reduce. This chapter focuses on the overview of software testing and current research problems. Additionally this chapter also highlights the work plan of the thesis.

1.2 OVERVIEW OF SOFTWARE TESTING

Software testing is a process which ensures that the delivered software is error free (as much as possible) and meets the software requirements. It is a routine work often asked by the stakeholders and end users. The process makes sure that the software is performing as expected. Along with software coding, software testing costs around 40%-50% of total software development cost (Klaib 2009). In system development, the life cycle can be considered as follows (Wikipedia, 2012)

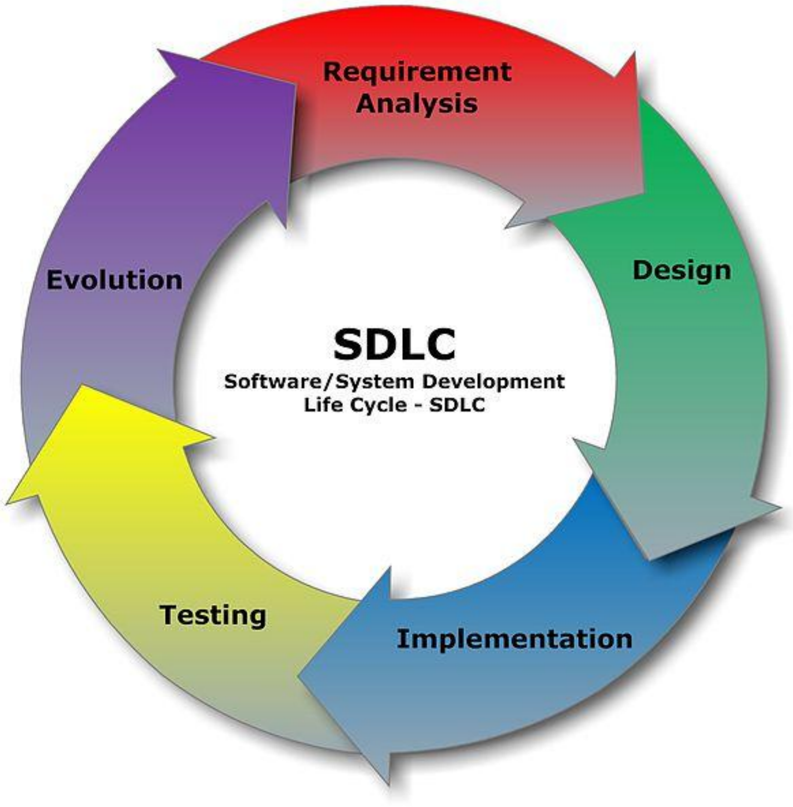


Figure 1.1: Software Engineering Product Life Cycle

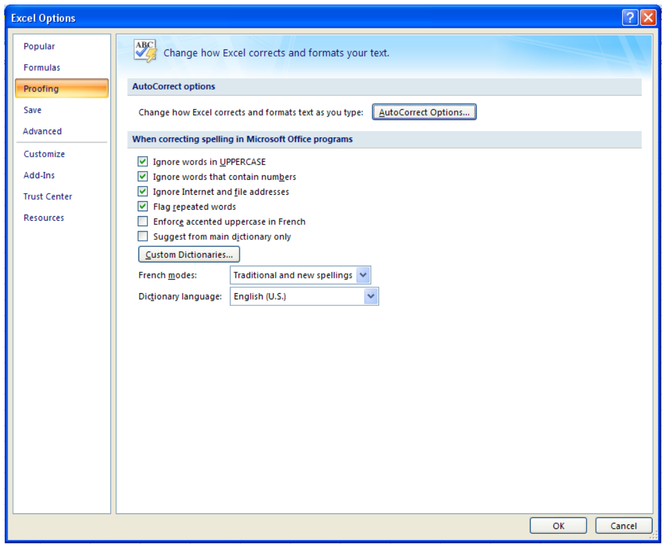
Referring to Figure 1.1, the system development starts with “Requirement Analysis”. In this phrase the customer/stakeholders prepare software/system specification which is also known as requirement document. The requirement document is then reviewed or tested by a verification and validation (V & V) team. After that the design of entire system including test plans is produces by managers or software architects. In the implementation phrase the software programmers usually write the line of codes and testers performs the code level testing which is also known as “Unit Testing”. In the testing phrase the all other functional or non functional testing drives by testers according the test plan. In this phrase the V & V team and/or customers performs an acceptance testing to make sure the software meets the specifications. And then the software is ready to release. In the evolution phrase maintenance testing is performed.

The testing itself don’t produce anything but it reduces the risks of the software hence improve the quality of the software and makes sure that the software is performing as it is expected. To perform the tests in different phrase, producing “Test Data” also known as “Test Cases” is one of the biggest challenges in the industry.

1.3 PROBLEM STATEMENTS AND MOTIVATION

Software testing and debugging is one of the integral part of software development life cycle in software engineering but this process is still very labour-intensive and expensive. Around 50% of project money goes under software testing (Klaib, 2009) Hence, the focus is to find automatic and cost-effective software testing and debugging techniques to ensure high quality of released product. Nowadays, research on software testing focuses on test coverage criterion design, test-data generation problem, test oracle problem, regression testing problem and fault localization problem. Among these problems test-data generation problem is an important issue in producing error free software. To solve this problem, Pairwise strategy (i.e. two-way interaction) has been known as an effective test data reduction strategy and able to detect from 60 to 80 % of the faults (Klaib, 2009).

For example, the ‘**proofing**’ tab under ‘**option**’ dialog in Microsoft excel (Figure 1.2), there are 6 possible configurations needed to be tested. Each configuration takes two values (checked or unchecked).Top of that the ‘**French modes**’ takes 3 possible values and ‘**Dictionary language**’ takes 54 possible values. So to test this proofing tab exhaustively, the number of test data need to be executed is 26 x 54 x 3 i.e. 10,368. Assuming each test data may consume 4 minutes to execute; results around 28 days to complete the exhaustive test of this ‘**proofing**’ tab (Klaib, 2009).



Change how Excel corrects and formats your text.

AutoCorrect options

Change how Excel corrects and formats text as you type:

When correcting spelling in Microsoft Office programs

- ✓ Ignore words in UPPERCASE
- ✓ Ignore words that contain numbers
- ✓ Ignore Internet and file addresses
- ✓ Flag repeated words
- ☐ Enforce accented uppercase in French
- ☐ Suggest from main dictionary only

French modes: Traditional and new spellings

Dictionary language: English (U.S.)

Figure 1.2: Microsoft Excel Proofing Option

Similar to a hardware product which has 30 on/off switches can give a considerable example. To test all possible combination may need 2^{30} = 1,073,741,824 test data, and consume 10,214 years by considering 5 minutes for each single test data. Nowadays, research work in combinatorial testing aims to generate least possible test data. The solution of this problem is non-deterministic polynomial-time hard (NP-hard) (Colbourn *et al.* 2004, Tai and Lei, 2002, Williams *et al.* 1996, Kuhn, D. R. *et al.* 2004). So far many approaches have been proposed and also many tools have been developed to find out the least possible test suit in polynomial time.

The examples shown above are known as "Combinatorial Explosion Problem" in common software testing. Since the time and resources are limited hence the common research questions are:

1. What is the smaller and optimal test data set?
2. What strategy should be chosen for test data generation?
3. What coverage should be chosen for testing the software?

In modern computing combinatorial explosion problem (Cohen *et al.* 1997, Cohen *et al.* 2006b, Colbourn *et al.* 2004, Tai and Lei, 2002, Klaib 2009) is known as one of the greatest challenge. From the last decade scientist from different place worked on this problem and proposed different strategy and methods to produce test data. Although through parallel testing (e.g. (ITL / NIST, 2008, Yunis *et al.*) 2009) the time can be reduce significantly. Since the complexity and width of the software getting bigger by days, it is also becoming impossible to do the exhaustive testing. International Software Testing and Quality Board (ISTQB) stated that "Exhaustive testing is impossible". Thus there should have strategies which help to produce quality test data (which is possible to execute) by keeping the standard quality.

Some researchers including Bryce and Colbourn, 2006, Dalal *et al.,* 1999, Kuhn and Okum, 2006, Kuhn and Reilly, 2002, Kuhn *et al.,* 2004, Yan and Zhang, 2008 stated that the interaction among different variables causes the hardware and software failures and the interaction is relatively very small (usually 3 to 6). If "T" is known as the interaction which may cause the failures the test data should be T-Way combination (Ellims *et al.,* 2008b). This T-Way combination also results the smaller number of test data. This T-Way combination is also known as "T-Way Testing". If the variable "T" is 2, it is known as 2-Way testing or "Pairwise Testing". The concept of "Pairwise

Testing" and "T-Way testing" is same differencing in pairwise T = 2 only and in T-Way T > 2.

Researchers developed lot of useful T-Way testing strategies to reduce the number of test data (i. e. to solve combinatorial explosion problem) including TConfig (Williams, 2000), CTS (Hartman and Raskin, 2004a), AETG (Cohen *et al*., 1997, Cohen *et al*., 1994), mAETG (Cohen, 2004, Cohen *et al.*, 2008, Cohen *et al.*, 2006a, Cohen *et al.*, 2007a, Cohen *et al.*, 2007b, Cohen *et al.*, 2003), GA (Shiba *et al.*, 2004), ACA (Shiba *et al*., 2004), IPOG (Lei *et al.*, 2007c), Jenny (Jenkins, 2003), TVG (Schroeder *et al.*, 2002, Schroeder *et al.*, 2003, Schroeder and Korel, 2000b), GTWay (Klaib *et al.*, 2009) and PSTG (Bestoun *et al.*,2010)). All these strategies use real data or some data models to produces test data. The data then converted into a special format to execute those against the system. However the test data generation time can be improved and automatic execution support can be included.

However, researchers find that producing minimum set of test data is NP-complete (Shiba *et al.*, 2004, Tai and Lei, 2002, Wan-qiu *et al.* 2012) (Not deterministic polynomial hard). More elaborately any specific strategy cannot always produce minimum number of test data with minimum generation time. By motivated on these problems this research investigates and developed several optimal strategies including PS2Way, EasyA, R2Way and MTTG. Unlike others, MTTG is a commercial prototype tool which also supports automatic execution for web based software systems. These strategies are useful for detecting software fault in a polynomial time which is the main hypothesis of this thesis.

1.4 RESEARCH AIM AND OBJECTIVES

The aim of this research is to investigate, develop and validate simpler and efficient pairwise and T-way test data generation and execution strategies. The main objectives of the work undertaken are:

1. To study and investigate test data generation for optimum solution.
2. To design an efficient T-Way strategy for data generation.
3. To develop and implement the T-Way strategy as a prototype tool.
4. To evaluate the performance of T-Way strategy in terms of test size and execution time.

1.5 SCOPE OF THE RESEARCH

The main issues regarding software testing is test data generation and execution. Typically it takes long time to generate test data and the execution process is still very laborious as most the strategies don't support auto execution.

In this thesis, four strategies have been proposed to overcome those issues. Finally a strategy has been developed to support auto execution. The scope of the research is to make a suitable strategy which supports both uniform and non-uniform values, generates test data based on real values (real input values used i.e. no index type values) and also able to support auto execution by developing a tool towards a smooth software testing process. The scope of the research is to develop a series of test data generation strategy and finally development of an efficient T-Way test data generation strategy. The proposed strategies also used dummy data to evaluate its result with existing strategies since the consideration and assumption has been taken from Klaib *et. al.* 2009 that has been used dummy data to convey his evaluation. The research supports testing in "Black box" testing phrase thus any type of product prior release to market can be tested through the proposed strategies is also the scope of the research.

1.6 STUDY MODULE

Figure 1.3 shows the summary of the research direction considered in this thesis. The followed direction to achieve the objectives is presented in an inner box, where the bold area means the research area which is not covered in this thesis.

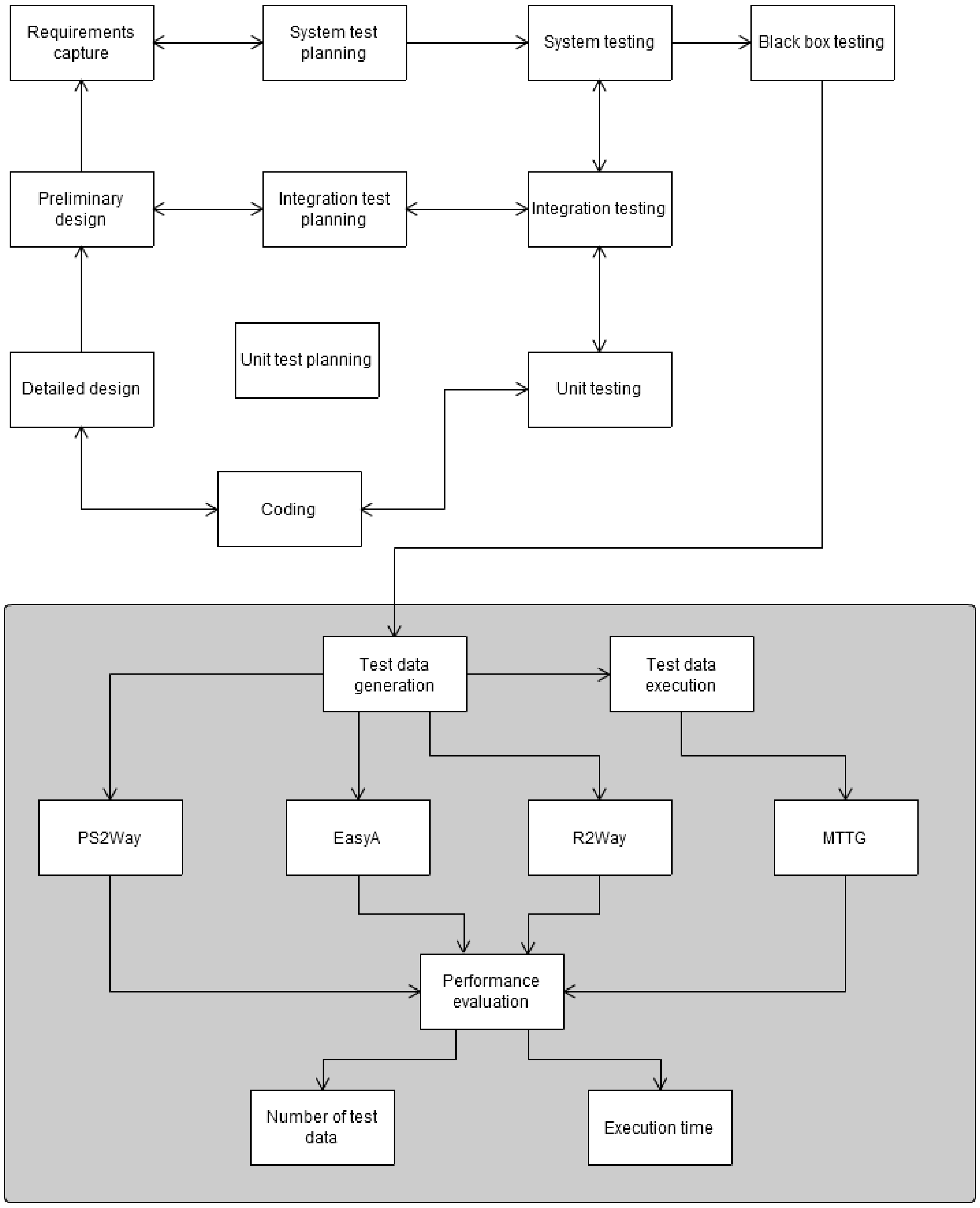


Figure 1.3: Study module

1.7 ORGANIZATION OF THESIS

The remainder of this thesis is organised into seven chapters as follows.

Chapter 2 presents an overview as well as highlights the main characteristics of 2 way and T-way strategies. At the end of Chapter 2, an analysis of existing works is presented which provides the requirements and justification for the development of a new strategy.

Chapter 3 discusses and justifies the detailed algorithm and implementation of pair parameter search based test data generation (PS2Way) strategy along with comparative results.

Chapter 4 discusses and justifies the detailed algorithm and implementation of matrix based strategy to generate 2-way test data (EasyA). The details description of the algorithm and the comparative result has also been shown.

Chapter 5 discusses and justifies the detailed algorithm and implementation of Random search based 2 way strategy (R2Way). The details description of the algorithm, the tool and the comparative result has been shown here.

Chapter 6 discusses and justifies the detailed algorithms and implementation of Multi-tuple based T-way test data generation strategy (MTTG) along with the tool and the comparative results.

Finally, the conclusion of this work is presented in Chapter 7 by highlighting the significance of findings along with considerations for future work.

CHAPTER 2

LITERATURE REVIEW

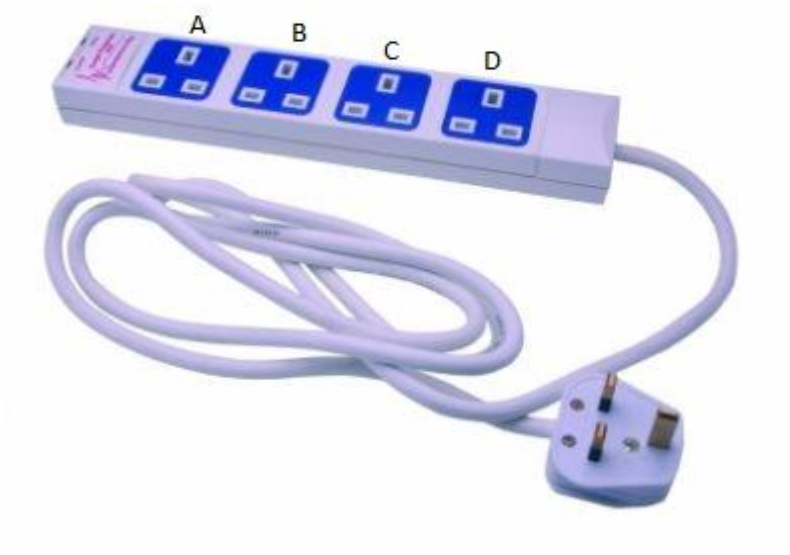
2.1 INTRODUCTION

In this chapter, some of the systematic approaches have been elaborated based on the T-way interaction of variables. It begins with an overview of concepts and terminology used in this thesis. Then, the main characteristics of combinatorial strategies are identified to facilitate their review and analysis. These analyses are then used to provide justification for the development of all the proposed strategies. Finally, this chapter ends with a brief summary.

2.2 OVERVIEW

One of the main objectives of this research is to minimize the test data based on the variable “T”. Here the variable “T” indicates the interaction among different input values. To understand the interaction “T” which also reflects the reduction of test data, following example is illustrated.

Figure 2.1: Example device - 4 point socket.



Referring Figure 2.1, testing a very simple 4 point socket has 4 sockets (input parameters) and each socket have on and off (input values). Thus it can be illustrated to the system as shown in Table 2.1. To identify the input parameters A, B, C and D has been named. Each parameter has on and off values.

Table 2.1: Parameter and Value illustration

A	**B**	**C**	**D**
On	On	On	On
Off	Off	Off	Off

Here, one test data or test case resides in one of the set of X= {A, B, C, D}. So all the possible sets of the system can be represent in Table 2.2 which is known as full power i. e. T = 4, also known as exhaustive numbers. Here, the numbers of all possible combination can also be calculated with a very easy formula [the number of test data = (number of values) $^{\text{the number of parameters}}$ = 2^4 = 16]. By clearly observing the data shown in the Table 2.2, it can also be seen that parameter A has repeated entry 16/2 = 8 times, where B has 8/2 = 4 times, C has 4/2 = 2 times and finally D has 2/2 = 1 times.

Table 2.2: Exhaustive Combinations (at T=4)

X (Number of all possible set)			
A	**B**	**C**	**D**
On	On	On	On
On	On	On	Off
On	On	Off	On
On	On	Off	Off
On	Off	On	On
On	Off	On	Off
On	Off	Off	On
On	Off	Off	Off
Off	On	On	On
Off	On	On	Off
Off	On	Off	On
Off	On	Off	Off
Off	Off	On	On
Off	Off	On	Off
Off	Off	Off	On
Off	Off	Off	Off

2.2.1 Don’t Care Formula

One basic strategy to reduce the test data is known as “Don’t Care”. From the above Table 2.2, if parameter D is known to have less impact on the system then it can be considered as “Don’t Care”. Thus, the value of D can be randomly anything. So the combination of ABC will be known as a 3-Way combination (T = 3). The combination of ABC has shown in Table 2.3.

Using this strategy the number of total test data (X) can be reduce from 16 to 8, i.e. 50% of reduction shown in Table 2.3. But most of the time it is unknown about the less impact parameter of a system. Almost every parameter has same effect to produce defects. Thus, it is possible to take all the combinations each in one time where each value can be “Don’t Care” one time which has been illustrated in Table 2.4.