AUTOMATED PAIRWISE TESTING APPROACH BASED ON CLASSIFICATION TREE MODELING AND NEGATIVE SELECTION ALGORITHM

EASTER VIVIANA ANAK SANDIN

A report submitted in partial fulfilment of the
requirements for the award of degree of
Master of Philosophy

School of Computing
Faculty of Engineering
Universiti Teknologi Malaysia

MARCH 2019

This report is dedicated with the deepest gratitude especially to my beloved family, as they always give me endless pray, love, support and motivation. To my friends who are willing to help me, and to all lecturers, especially my supervisors, for without their early inspiration, coaching and enthusiasm none of this would have happened. Thank you for everything.

ACKNOWLEDGEMENT

Praise be to God, I have finally completed the research proposal for my Master research titled "*Pairwise Testing Approach Based on Classification Tree Modeling and Negative Selection Algorithm*".

I would like to extend my sincerest gratitude to my supervisor, Associate Prof. Dr. Radziah binti Mohamad for her excellent guidance and her vast experience towards the completion of this report. She had been very helpful and resourceful from the very beginning until the end of this thesis.

Endless thanks go to my friends who had involved themselves in the completion of my report, especially those who are willing to spare their precious time in helping me whenever requested. Their supports had inspired me along the way to finish my report.

Lastly, my gratitude goes to those who had involved themselves directly or indirectly in the completion of this report.

ABSTRACT

Generating the test cases for analysis is an important activity in software testing to increase the trust level of users. The traditional way to generate test cases is called exhaustive testing. It is infeasible and time consuming because it generates too many numbers of test cases. A combinatorial testing was used to solve the exhaustive testing problem. The popular technique in combinatorial testing is called pairwise testing that involves the interaction of two parameters. Although pairwise testing can cover the exhaustive testing problems, there are several issues that should be considered. First issue is related to modeling of the system under test (SUT) as a preprocess for test case generation as it has yet to be implemented in automated proposed approaches. The second issue is different approaches generate different number of test cases for different covering arrays. These issues showed that there is no one efficient way to find the optimal solution in pairwise testing that would consider the invalid combination or constraint. Therefore, a combination of Classification Tree Method and Negative Selection Algorithm (CTM-NSA) was developed in this research. The CTM approach was revised and enhanced to be used as the automated modeling and NSA approach was developed to optimize the pairwise testing by generate the low number of test cases. The findings showed that the CTM-NSA outperformed the other modeling method in terms of easing the tester and generating a low number of test cases in the small SUT size. Furthermore, it is comparable to the efficient approaches as compared to many of the test case generation approaches in large SUT size as it has good characteristic in detecting the self and non-self-sample. This characteristic occurs during the detection stage of NSA by covering the best combination of values for all parameters and considers the invalid combinations or constraints in order to achieve a hundred percent pairwise testing coverage. In addition, validation of the approach was performed using Statistical Wilcoxon Signed-Rank Test. Based on these findings, CTM-NSA had been shown to be able perform modeling in an automated way and achieve the minimum or a low number of test cases in small SUT size.

ABSTRAK

Menghasilkan kes ujian adalah aktiviti penting dalam pengujian perisian untuk meningkatkan tahap kepercayaan pengguna. Cara tradisional untuk menghasilkan kes ujian ialah ujian menyeluruh. Ujian ini sukar untuk dilaksanakan dan menelan masa yang banyak kerana menghasilkan banyak nombor kes ujian. Ujian kombinasi telah diwujudkan untuk menyelesaikan masalah ujian menyeluruh. Teknik ujian kombinasi yang digemari adalah ujian berpasangan yang melibatkan interaksi antara dua parameter. Walaupun ujian kombinasi mengatasi masalah ujian menyeluruh, namun terdapat beberapa isu yang perlu diambilkira. Isu pertama ialah berkaitan dengan permodelan sistem di bawah ujian (SUT) sebagai pra proses untuk penghasilan kes ujian secara automatik. Isu kedua adalah pendekatan berbeza menghasilkan bilangan kes ujian yang berbeza bagi tatasusunan yang berlainan. Isu ini menunjukkan bahawa tiada cara yang efisien untuk mencari penyelesaian optimum yang juga mempertimbangkan gabungan atau kekangan yang tidak sah. Oleh itu, kombinasi Kaedah Pokok Klasifikasi dan Algorithma Pemilihan Negatif (CTM-NSA) telah dibangunkan dalam kajian ini. Pendekatan CTM telah dipelajari dan dipertingkatkan untuk dijadikan permodelan automatik dan pendekatan NSA dibangunkan untuk mengoptimumkan ujian berpasangan. Hasil kajian mendapati bahawa CTM-NSA dapat mengatasi kaedah model lain dalam menyenangkan penguji dan menghasilkan sedikit bilangan kes ujian untuk saiz SUT kecil dan juga setanding dengan pendekatan lain dalam saiz SUT besar kerana mempunyai ciri-ciri mengesan sampel diri dan bukan diri. Ciri-ciri ini berlaku di peringkat pengesanan NSA yang merangkumi kombinasi nilai-nilai terbaik bagi semua parameter dengan menganggap kombinasi atau kekangan yang tidak sah dalam mencapai 100 peratus liputan ujian berpasangan. Pengesahan pendekatan ini menggunakan Ujian Statistik Wilcoxon Signed-Rank. Berdasarkan hasil kajian ini, CTM-NSA mampu melakukan pemodelan secara automatik dan menghasilkan kes ujian minimum atau rendah untuk saiz SUT yang kecil.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABC	-	Ant Bee Colony Algorithm
ACA	-	Ant Colony Algorithm
API	-	Application Programming Interface
AETG	-	Automatic Efficient Test Generator
SAT	-	Boolean Satisfiability Solving
CPM	-	Classification Parameter Method
CTM	-	Classification Tree Method
CT	-	Combinatorial Testing
CA	-	Covering Array
CS	-	Cuckoo Search Algorithm
DDA	-	Deterministic Density Algorithm
FS	-	Flower Pollination Algorithm
GA	-	Genetic Algorithm
GS	-	Genetic Strategy
HD	-	Hamming Distance
HS	-	Harmony Search Algorithm
HHH	-	High Level Hyper-Heuristic
HC	-	Hill Climbing Algorithm
HOA	-	Hybrid Optimization Approach
IPO	-	In-Parameter-Order
IPM	-	Input Parameter Modeling
OPAT	-	One-parameter-at-a-time
OTAT	-	One-test-at-a-time
PICT	-	Pairwise Independent Combination Testing
PSO	-	Particle Swarm Optimization
SA	-	Simulated Annealing Algorithm
SDLC	-	Software Development Life Cycle

SUT	-	System Under Testing
TCG	-	Test Case Generator
UML	-	Unified Modeling Language

LIST OF SYMBOLS

$\in$	-	Element of
$\sum$	-	Sum of
$<$	-	Less than
$\oplus$	-	XOR or Exclusive OR
τ	-	Threshold
μ	-	Mean
α	-	Alpha

CHAPTER 1

INTRODUCTION

1.1 Overview

Nowadays, the rapid developments of intelligence technologies grow as the demand grows. They put their trust on those technologies. For example, the web system such as food delivery website let people order their meal through the website without going to the restaurant. Technology has made life simpler and more convenient because this matter will save their time when doing important work. For an embedded system such as an airplane system, 100% assurance is needed as they are used to carry many lives in them. However, the question is how many people can put their trust on those technologies? Therefore, software testing is one of the important activities that should be performed in order to gain and present the software trustworthiness.

Software testing consists of black box testing and white box testing (Khalsa and Labiche, 2014). A black box is focused on external behavior or functionality while a white box is focused on internal implementation of software. In order to conduct software testing, the test cases should be ready first. This activity falls into black box testing where it involves the specification only. The traditional way to generate the test cases is called exhaustive testing. Exhaustive testing is used to produce the test suite that will be used in other testing types such as unit testing, system testing, integration testing and acceptance testing. The example of how to conduct the exhaustive testing is as following; Assume that the parameters are A, B, and C. The values are as stated; A= (a1, a2), B= (b1, b2), C= (c1, c2, c3). The number of test cases generated through this method will be 2x3x2= 12 tests; as shows in the figure below.

A	B	C
a1	b1	c1
a1	b1	c2
a1	b1	c3
a1	b2	c1
a1	b2	c2
a1	b2	c3
a2	b1	c1
a2	b1	c2
a2	b1	c3
a2	b2	c1
a2	b2	c2
a2	b2	c3

Figure 1.1 Generated test cases for exhaustive testing

However, the issues or problems with exhaustive testing are discovered when it comes to large or complex software systems. The popular issue of exhaustive testing is costly and time-consuming (Purohit and Khan, 2015). Imagine if this research has a large number of parameters and values, it may generate about thousands of test cases. Therefore, the combinatorial testing (CT) is proposed to solve the exhaustive testing problem.

CT is the black box type of testing (Brcic and Kalpic, 2012) (Mudarakola and Padmaja, 2015). It can provide a better way for test cases generation. It can reduce the cost of testing and save the testing time to increase its effectiveness (Borazjany et al, 2013); (Kitamura et al, 2015); (Nie and Leung, 2011); (Brcic and Kalpic, 2012); (Patil and Nikumbh, 2012). There are many techniques involve in CT. CT consists of one technique that is called t-way testing. This technique is a popular research area among researchers (Kitamura et al, 2015). It requires all combinations of values of t-parameter that are at least tested once. There are six types of t-way testing, which are 1-way, 2-way, 3-way, 4-way, 5-way and 6-way (Kuhn et al, 2013). Among these t-way types, 2-way is the wild technique in CT problems (Mudarakola and Padmaja, 2015) (Bach and Schroeder, 2004). 2-way testing is called Pairwise Testing. It is used to decrease the number of test cases or test suite generated, where it considers all interaction of two factors the most (Xiang et al, 2015). This means that they detect the constraint or problem between the interactions of two parameters. The aim of this pairwise testing is to cover every pair of options in testing. Every pair of options must occur at least once and may occur more than once (Kuhn, 2013). The other advantages of pairwise testing are easy to manage and executed by testers (Bach and Schroeder, 2004).

1.2 Problem Background

Pairwise testing is a test case generation technique that is caused by the interaction of two parameters-values. It covers the combination of two parameters-values, therefore it generates the lower number of test cases compared to exhaustive testing. Pairwise testing has its own procedures to perform it (Nie and Leung, 2011). In order to generate the test case, the modeling for SUT should be first performed as a pre-process for it. It is a fundamental activity for pairwise testing as the precise model will serve the right level of abstraction (Udai, 2014).

The quality of pairwise testing is directly dependent on the quality of the model created (Staich and Rangarajan, 2016); Borazjany et al, 2013). This is because the systematic model will cover the problem of managing the SUT information, especially for a large system. One of the examples of the existing problem such as incomplete data or manageable (that affect the time and cost of testing) (Khalsa and Labiche, 2015). The information of SUT might be redundant as the input of test cases generator. The incomplete input of test case generation refers to some of the information which is left unwritten (missing information), while unmanageable refers to the "messy" values of the parameter (lead to a wrong place of value). Encountering the issue of unmanageable will make the program unable to detect the failure of a system after generating the test case.

The flow of pairwise testing is manageable and understandable if the model can be embedded with test case generation approach. The updating of the parameters and values can be performed through the model only without disturbing the hard code of test case generation algorithm. However, there is the lack of approaches that embedded the modeling of SUT with test case generation algorithm.

The finding for a low number of test cases is the NP-complete problem as there is no efficient way to find an optimal solution for it and the execution time to generate the test cases increased due to the number of parameters and values (Patil and Nikumbh, 2012). There is no best approach that can generate the test cases. Furthermore, the issue of invalid combinations of values for all parameters is also an important aspect to study. It can lead to faulty results in software testing. For example, a Vegetable Lover value cannot combine with the Fried Chicken value. It is obviously a wrong combination. This is also called as a constraint for pairwise testing. Some of the existing approaches still do not cover this matter.

There are many researchers who have conducted researches on pairwise testing and many approaches have been proposed from time to time (Khalsa and Labiche, 2014; Mudarakola and Padmaja, 2015; Parnami et al., 2012; Udai, 2014).

Test case generation for pairwise testing can be classified into several categories, namely mathematical approach, random approach, greedy approach, search-based approach and hybrid approach (Sabharwal and Aggarwal, 2015). However, each of these approaches has their own advantages and disadvantages. This will be discussed in detail in the next chapter.

For a general introduction, in the mathematical approach, the generation of test cases is based on the mathematical solution. Unfortunately, they are not generally applicable (Calvagna and Gargantini, 2009). Random search-based is producing the solution by depending on the degree of randomness of approaches. However, they did not cover the large or complex software system for pairwise testing (Khatun et al., 2011). Greedy approach is generating test cases by covering as many as possible the uncovered combinations. However, this category does not always cater for the optimal solution (Calvagna and Gargantini, 2009). A hybrid-approach is a combination of two or more approaches from any categories. The aim of this approach-based is to enhance the existing approaches by combining their advantages. However, these approaches may lead to high computational time.

A search-based approach is one of the most emerging technologies for the last 20 years (Nasser et al, 2015). This approach type applies the meta-heuristic algorithm to solve software engineering problems. It has been widely used in many activities of the software engineering lifecycle including in the test case generation for pairwise testing. One of the highlights about this approach type is its ability to find the minimal test suite (Nasser et al, 2015). The strategies for this approach type is divided into two; single-solution based and population-based. The single-solution based focuses on a local search where it only needs a limited amount of memory for execution. However, this strategy is stuck in the local optimum solution. On the other hand, population-based focuses on global search where it reaches the global optimum solution. However, it requires heavy computational effort. Therefore, it addresses a small configuration only (Harman and Jones, 2001).

Another issue that is related to the search based approach is there are prerequisites that need to be tuned (Nasser et al, 2015). For example, GA needs tuning of mutation rate, crossover rate, number of iteration and population size. Therefore, the researchers are contemplating the prerequisite free approaches for pairwise testing. Although there are many studies that have successfully adopted search-based approach for pairwise test case generation, there are many other algorithms for search based that have not been adopted in this area (Nasser et al, 2015).

Automating test case generation is a popular research topic that gains the interest of many researchers. Recently, search-based approaches are the most widely used methods in generating the test case automatically. Although there are several approaches recently proposed for automating the generation of the test case, the application of these approaches to find the optimal solution is still limited. The optimal test cases set are obtained if their generated number is low. Besides, there is a lack of automation approaches that embedded the modeling of SUT. Hence, performing a study and proposing a search-based approach that can cover the existing problems or issues in pairwise testing is needed.

1.3 Statement of the Problem

With the rapid development of technologies, many developers and testers tend to use automated test case generation. It can simplify their work and help them in terms of efficiency for the testing phase. Testing implementation using automated software is the best solution especially for those who have a poor command of programming languages and for beginner developers because it can be used for many purposes or functionalities. However, different approaches have different specific functionalities.

This research aims at investigating an automated-approach based on Negative Selection (NSA) in generating the test cases with minimal numbers. Before conducting the generation of test cases using NSA, the modeling of SUT should be performed first. Based on the statement in the previous paragraph, different model methods and test case generation approaches serve the different purposes and functionalities. Therefore with this issue, research questions as following are generated:

How to optimize the number of test cases for pairwise testing by using search-based algorithm?

i. How to enhance the modeling of SUT for pairwise testing?
ii. How to improve a search based algorithm for pairwise testing?
iii. How to validate the proposed approach on optimizing the pairwise testing?

1.4 Objectives

In order to achieve the goal of this research, 3 objectives have been defined. The objectives are:

i. To enhance the classification tree method for modeling of pairwise testing.
ii. To improve a search-based approach for optimizing the pairwise testing.
iii. To validate the proposed-approach toward optimizing the pairwise testing.

1.5 Significance of the Research

The task of this research is to optimize the pairwise testing by reducing or produces or generates the minimal number of test cases. Reduce the number of test cases is important because low number of test cases to be executed lead to reducing the total testing time (Borazjany et al, 2013). In order to perform the pairwise testing, there are 2 things should be considered. Firstly, the modeling of SUT should be done before generate the test cases (Udai, 2014). The second important activity in pairwise testing is generating the test cases (Nie and Leung, 2011). Hence, the study that is related to the existing works for these 2 activities can be done to propose the approach to optimizing pairwise testing.

1.6 Scope of the Research

Although there are many techniques in combinatorial testing, this research is focused on 2-way or pairwise testing only. It is the most popular technique in combinatorial testing (Mudarakola and Padmaja, 2015). Since pairwise testing is a black box testing, hence this research is only considering the modeling of SUT in black box testing categories. Moreover, this research has proposed an algorithm that falls under search based type for test case generation. Lastly, this research only uses the experimental data set to compare and analyze the result of the proposed approach to achieve the optimizing pairwise testing goal.

1.7 Organization of Thesis

A brief content description of the subsequent chapter is summarized as below: Chapter 1 introduces the concept of this research in detail. It discusses the background of the problem, statement of the problem, objectives, significant of the study and an organization of thesis. Chapter 2 presents the introduction of pairwise testing, and related works on this topic are also presented in this chapter. Chapter 3 presents the detailed description of the research workflows, which includes the research framework and design. Chapter 4 presents the implementation of the proposed approach. Chapter 5 discusses the analysis for result gained in Chapter 4 and Chapter 6 discusses the future work and conclusion.

REFERENCES

Ahmed, B. S., Abdulsamad, T. S., and Potrus, M. Y. (2015). *Achievement of Minimized Combinatorial Test Suite for Configuration-Aware Software Function Testing Using The Cuckoo Search Algorithm.* Information and Software Technology 66 (2015) 13-29.

Ahmed, B. S., and Zamli, K. Z. (2011). *The Development of A Particle Swarm Based Optimization Strategy for Pairwise Testing.* Journal of Artificial Intelligence 4 (2): 156-165, 2011.

Alazzawi, A. K., Homaid, A. A. B., Alomouch, A. A., and Alsewari A. A. (2017). *Artificial Bee Colony Algorithm for Pairwise Test Generation*. Journal of Telecommunication, Electronic and Computer Engineering. vol. 9 no. 1-2.

Alsewari, A. R., Zamli, K. Z., and Kazemi, B. A. (2015). *Generating T-way Test Suite In The Presence of Constraints.* Journal of Engineering and Technology. Vol. 6 No. 2.

Alsewari, A. R., and Zamli, K. Z. (2012). *Design and Implementation of A Harmony-Search-Based Variable-Strength T-way Testing Strategy With Constraints Support.* Information and Software Technology 54 (6), Pp. 553-568.

Alsewari, A. R., and Zamli K. Z. (2011). *Interaction Test Data Generation Using Harmony Search Algorithm.* IEEE Symposium on Industrial Electronics and Application (ISIEA2011), September 25-28, 2011, Langkawi, Malaysia

Bach, J., and Schroeder, P. J. (2004). *Pairwise testing - A best practice that isn't.* In Proceedings of the 22nd Pacific Northwest Software Quality Conference. 180–196.

Bach, J. (2003). *Allpairs.* Retrieved on January 2018. Available online: Satisfice.com

Bansal, P., Mittal, N., Sabharwal, A., and Koul, S. (2014) *Integrating Greedy Based Approach with Genetic Algorithm to Generate Mixed Covering Arrays* *for Pair-wise Testing.*The 7th International Conference on Contemporary Computing, Noida.

Bansal, P., Sabharwal, S., Malik, S., Arora, V. and Kumar, V. (2013) *An Approach to Test Set Generation for Pair-Wise Testing Using Genetic Algorithms.* Search Based Software Engineering, Springer Berlin Heidelberg, Pp. 294-299.

Bao, X., Liu, S., Zhang, N., and Dong, M. (2015). *Combinatorial Test Generation Using Improved Harmony Search Algorithm.* International Journal of Hybrid Information Technology Vol. 8, No. 9, Pp. 121-130.

Boehm, B. W. (1984). *Understanding and Controlling Software Costs*. IEEE Transactions on Software Engineering. Vol. 14 (Issue 10), Page 1462-1477.

Borazjany, M. N., Ghandehari, L. S., Lei, Y., Kacker, R., and Kuhn, R. (2013) *An input space modeling methodology for combinatorial testing*. In Software Testing, Verification and Validation Workshops (ICSTW), 2013 IEEE Sixth International Conference on, pp. 372-381.

Brcic, M., and Kalpic, D. (2012) *Combinatorial testing in software projects*. In MIPRO, 2012 Proceedings of the 35th International Convention, pages 1508–1513, 2012.

Byrce, R. C., and Colbourn C. J. (2007). *The density algorithm for pairwise interaction testing.* Software Testing, Verification and Reliability (in Press).

Calvagna, A., and Gargantini A. (2009). *IPO-s: Incremental generation of combinatorial interaction test data based on symmetries of covering arrays.* IEEE International Conference on Software testing Verification and Validation Workshops.

Chen, T. Y., Poonb, P.L., Tang, S. F., Tse, T.H. (2004*). On the identification of categories and choices for specification-based test case generation*. Article in Information and Software Technology 46 (13): 887–898

Chen, X., Gu, Q., Qi, J., and Chen, D. (2010). *Applying Particle Swarm Optimization to Pairwise Testing*. 34th Annual IEEE Computer Software and Applications Conference. China. 2010. Pp 107-116

Cohen, M. B., Gibson, P. B., Mugridge, W. B., and Colbourn, C. J. (2003). *Conctructing Test Suites for Interaction Testing.* Proc. Of the 25th International Conference on Software Engineering (ICSE '99), pp. 285-94.

Cohen, M. B. (2004). *Designing Test Suites for Software Interaction Testing.* PHD Thesis. University of Auckland.

Cui, Y., Li, L., and Yao, S. (2009). *A New Strategy for Pairwise Test Case Generation.* 3rd International Symposium on Intelligent Information Technology Application.

Czerwonka, J. (2016). *Pairwise Testing Combinatorial Test Case Generation.* Retrieved on 16 March 2016. www.pairwise.org

Dasgupta, S., and Nino, F. (2008). *Immunological Computation: Theory and Applications,* USA: CRC Press.

Do, T. B., Kitamura, T., Tang, N. V., Hatayama, G., Sakuragi, S., and Ohsaki, H. (2013) *Constructing test cases for N-wise testing from tree-based test models*. In Proc. of SoICT'13, pages 275–284. ACM.

Easterbrook, S., Sim, S., Perry, D., Aranda, J. (2006) *Case Studies for Software Engineers Tutorial*. Proc. Of ICSE´2006, Shanghai, Chin, May 2006.

Esfandyari, S., and Rafe V. (2018). *A Tuned Version of Genetic Algorithm for Efficient Test Suite Generation in Interactive T-way Testing Strategy.* Information and Software Technology 94 (2018) 165-185

Fan, P., Wang, S. and Sun, J. (2012). *An Auto-Adapted Method to Generate Pairwise Test Data Set.* Artificial Intelligence and Computational Intelligence, Springer Berlin Heidelberg, Pp. 239-246.

Flores, P., and Cheon, Y. (2011). *PWiseGen: Generating Test Cases for Pairwise Testing Using Genetic Algorithms.* IEEE International Conference on Computer Science and Automation Engineering (CSAE), Vol. 2, Pp. 747-752

Forrest, S., Alan S. P., Lawrence A., and Rajesh C. (1994). *Self-Nonself Discrimination in a Computer*. Proceedings of the 1994 IEEE Symposium on Security and Privacy, 16-18 May. Oakland, CA, USA: IEEE, Pp. 202-212.

Gao, S., Du, B., Jiang, Y., Lv, J., and Ma, S. (2014). *An Efficient Algorithm for Pairwise Test Case Generation in Presence of Constraints.* 2nd International Conference on Systems and Informatics. China. Pp 406-410.

Getty, S. R., Wilson C. D., Taylor J. A., and Kowalski S. M. (2002). *Managing Threats to Validity in Experimental Tests of Education In terventions: Data and Evidence from A Large, Cluster-Randomized Trial (CRT) of A High School Science Intervention.* A Science Education Curriculum Study.

Grochtmann, M., Joachim, W., and Klaus, G. (1995) *Test case design using classification trees and the classification-tree editor CTE.* In Proceedings of Quality Week, vol. 95, p. 30.

Harman, M., and Jones, B. F. (2001) *Search-based Software Engineering.* Information and Software Technology. Vol 43, Pp. 833-839

Idris, I., Selamat, A., Nguyen, N. T., Omatu, S., Krejcar O., Kuca, K. and Penhaker, M. (2015). *A Combined Negative Selection Algorithm- Particle Swarm Optimization For An Email Spam Detection System.* Journal of Engineering Applications of Artificial Intelligence 39 Pp.33-44.

Idris, I., Selamat, A. and Omatu, S. (2014). *Hybrid Email Spam Detection Model With Negative Selection Algorithm and Differential Evolution.* Engineering Applications of Artificial Intelligence 28, Pp. 97-110.

Jenkins, B. (2005). Jenny Download Web Page. Retrieved on October 2017. Available online: http://burtleburtle.net/bob/math/jenny.html

Jeong, O. (2012). *A Practical Extension of Pairwise Testing.* 23rd International Symposium on Software Reliability Engineering Workshops.

Jia, M., and Shengyuan, W. (2014). *An Improved Genetic Algorithm for Test Cases Generation Oriented Paths.* Chinese Journal of Electronics Vol. 23, No. 3.

Ji, Z., and Dasgupta, D. (2007). *Revisiting Negative Selection Algorithms.* Evolutionary Computing 15(2): Pp. 223-251.

Khalsa, S. K., and Labiche, Y. (2014) *An orchestrated survey of available algorithms and tools for combinatorial testing*. in 25th IEEE International Symposium on Software Reliability Engineering, ISSRE 2014, Naples, Italy, November 3-6, 2014, 2014, pp. 323–334.

Khatun, S., Rabbi, K.F., Yaakub, C.Y., and Klaib, M.F. (2011) *A Random Search Based Effective Algorithm for Pairwise Test Data Generation*. International Conference on Electrical, Control and Computer Engineering. 2011. Malaysia and Jordan.

Kitamura, T., Yamada, A., Hatayama, G., Artho, C., Choi, E. H., Do, N. T. B., Oiwa, Y., Sakuragi, S. (2015). *Combinatorial Testing for Tree-Structured Test Models with Constraints.* 2015 IEEE International Conference on Software Quality, Reliability and Security. Pp 141-150

Kobayashi, N., Tsuchiya, T., and Kikuno, T. (2002). *Non-specification-based Approaches to Logic Testing for Software.* Journal of Information and Software Technology 44 (2), Pp. 113-121

Kuhn, R. (2016). *Combinatorial Methods in Software Testing.* Conference on Applied Statistic in Defense. October 26.

Kruse, P. M. (2013). *Enhanced Test Case Generation with the Classification Tree Method.* PHD Thesis

Kruse, P. M. and Wegener, J. (2012) *Test Sequence Generation from Classification Trees* in Proceedings of ICST 2012 Workshops (ICSTW 2012), Montreal,Canada, April 2012

Khatun, S., Rabbi, K. F., Yaakub, C. Y., and Klaib, M. F. J. (2011). *A Random Search Based Effective Algorithm for Pairwise Test Data Generation.* International Conference on Electrical, Control and Computer Engineering, Pahang, Malaysia, June 21-22.

Khatun, S., Rabbi, K. F., Yaakub, C. Y., Klaib, M. F. J. and Mohammad Masroor Ahmed (2011). *PS2Way: An Efficient Pairwise Search Approach for Test Data Generation.* J. M. Zain et al. (Eds.): ICSECS 2011, Part III, CCIS 181, Pp. 99-108. Springer-Verlag Berlin Heidelberg.

Kitamura, T., Yamada, A., Hatayama, G., Artho, C., Choi, E. H., Do, N. T. B., Oiwa, Y., and Sakuragi, S. (2015). *Combinatorial Testing for Tree-Structured Test Models with Constraints.* IEEE International Conference on Software Quality, Reliability and Security.

Kuhn, D. R., Kacker, R. N., and Lei, Y. (2013) *Introduction to Combinatorial Testing*, 1st ed. Chapman & Hall/CRC, London, UK.

Kumar, R. and Singh, S. (2010) *Breeding Software Test Cases for Pairwise Testing Using GA.* Global Journal of Computer Science and Technology, Vol. 10 Issue 4 Ver. 1.0.

Fan, P., Wang, S. and Sun, J. (2012) *An Auto-Adapted Method to Generate Pairwise Test Data Set*. Berlin. Springer-Verlag. 2012. Pp. 239-246

Lei, Y., and Tai, K. C. (1998). *In-Parameter-Order: A Test Generation Strategy for Pairwise Testing",* Proc. 3rd IEEE International Symposium on High Assurance Systems Engineering, pp. 254-261, Nov 1998.

Lei, Y., Kacker, R., Kuhn, D. R., Okun, V. and Lawrence, J. (2007) *IPOG: A General Strategy for T-way Software Testing.* In Proceedings of the 14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'07). IEEE Computer Society, Los Alamitos, CA, Pp. 549-556.

Li, X., Gao, R., Wong, W. E., Yang, C., and Li, D. (2016) *Applying Combinatorial Testing in Industrial Settings.* IEEE International Conference on Software Quality, Reliability and Security. 2016

Lott, C., Jain, A., and Dalal, S. (2005) *Modeling Requirements for Combinatorial Software Testing*. SIGSOFT Softw. Engin. Notes 30, 4, 1-7

Marcos, E. (2005) *Software engineering research versus software development*, ACM SIGSOFT Software Engineering Notes, v.30 n.4.

Mats, G,. and Offutt, J. (2007) *Input Parameter Modeling for Combination Strategies.* Proceedings of the 25th Conference on IASTED International Multi-Conference (SE'07). ACTA Press, 255-260

McCaffrey, J. D. (2010) *An Empirical Study of Pairwise Test Set generation using A Genetic Algorithm.* Seventh International Conference on Information Technology. 2010. Washington, United State. Pp 992-997

McCaffrey, J. D. (2009) *Generation of Pairwise Test Sets Using A Genetic Algorithm.* 33rd Annual IEEE International Computer Software and Applications Conference.

McCaffrey, J. D. (2009) *Generation of Pairwise Test Sets Using A Simulated Bee Colony Algorithm.* IEEE IRI 2009, July 10-12, 2009, Las Vegas, Nevada, USA.

Mills, D.L. (2008). *Testing for Web Applications.* USA: University of Memphis.

Mohd Ehmer Khan (2010) *Different Forms of Software Testing Techniques for Finding Errors*. IJCSI International Journal of Computer Science Issues,Vol. 7, Issue 3, No 1, May 2010.

Mohd Ehmer Khan & Farmeena Khan. (2012). A Comparative Study of White Box, Black Box and Grey Box Testing Techniques. International Journal of Advanced Computer Science and Applications, 3(6), 12–15. https://doi.org/10.1017/CBO9781107415324.004

Mohammod Abul Kashem and Mohammad Naderuzzaman (2013). *An Enhanced Pairwise Search Approach for Generating Optimum Number of Test Data and Reduce Execution Time.* Journal of Computer Engineering and Intelligent Systems, Vol. 4, No. 1.

Mudarakola, L. P., and Padmaja, M. (2015). *The Survey on Artificial Life Techniques for Generating the Test Cases for Combinatorial Testing.* International Journal of Research Studies in Computer Science and Engineering (IJRSCSE) Vol. 2, Issue 6. June 2015. Pp. 19-26

Mudarakola, L. P., Sastry, J., and Vudatha, C. P. (2014). *Generating Test Cases for Testing WEB Sites Through Neural Networks and Input Pairs.* International Journal of Applied Engineering Research Vol. 9, No. 22. Pp. 11819-11831.

Mustafa, Mohi-Aldeen S., Mohamad, R. and Deris, S. (2016). *Application of Negative Selection Algorithm (NSA) for Test Data Generation of Path Testing.* Applied Soft Computing Journal.

Nanba, T., Tsuchiya, T., and Kikuno, T. (2011) *Constructing Test Sets for Pairwise Testing: A SAT-Based Approach.* 2nd International Conference on Networking and Computing.

Nasser, A. B., Alsewari, A. A., and Zamli, K. Z. (2015) *Adopting Search-Based Algorithms for Pairwise Testing.* 4th International Conference on Software Engineering and Computer Systems (ICSECS), Kuantan, Pahang, Malaysia. August 2015.

Nasser, A. B., Sariera, Y. A., Alsewari, A. R. A., and Zamli, K. Z. (2015). *A Cuckoo Search Based Pairwise Strategy For Combinatorial Testing Problem.* Journal of Theoretical and Applied Information Technology. Vol. 82. No 1.

Nasser, A. B., Sariera, Y. A., Alsewari, A. R. A., and Zamli, K. Z. (2015). *Assessing Optimization Based Strategies for t-way Test Suite Generation: The Case for Flower-based Strategy.* IEEE International Conference on Control System, Computing and Engineering, 27-29 November 2015, Penang, Malaysia.

Nasser, A. B., Alsewari A. A, Tairan N. M. and Zamli K. Z. (2017). *Pairwise Test Data Generation Based on Flower Pollination Algorithm.* Malaysia Journal of Computer Science vol 30(3).

Nguyen, V. T., Nguyen, T. T., Mai, K. T. and Le, T. D. (2014). *A Combination of Negative Selection Algorithm and Artificial Immune Network for Virus Detection.* T. K. Dang et al. (Eds.): FDSE 2014, LNCS 8860, Pp. 97-1-6. Springer International Publishing Switzerland 2014.

Nidhra, S., and Dondeti, J. (2012) *Black Box and White Box Testing Techniques- A Literature Review.* International Journal of Embedded Systems and Applications (IJESA) Vol. 2, No. 2, June 2012.

Nie, C., and Leung, H. (2011). *A Survey of Combinatorial Testing.* ACM Computing Surveys, Vol. 43, No. 2, Article 11. January 2011.

Ostrand, T. J., and Balcer, M. J. (1988). *The Category-Partition Method for Specifying and Generating Functional Tests.* Communications of the ACM. Volume 32 (6), Pp. 676-686.

Patil, M., and Nikumbh, P. J. (2012). *Pair-wise Testing Using Simulated Annealing.* Proceedings 2nd International Conference on Computer, Communication, Control and Information Technology (C3IT-2012), Feb. 2012, Pp. 25-26.

Parnami, S., Sharma, K. S. and Chande, S. V. (2012). *A Survey on Generation of Test Cases and Test Data Using Artificial Intelligence Techniques.* Proceeding of the International Conference on Advances in Computer Science and Electronics Engineering.

Purohit, P., and Khan, Y. (2015) *An Automated Sequence Model Testing (ASMT) For Improved Test Case Generation Using Cloud Integration.* (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 6 (1) , 2015, 488-494

Qi, R., Wang, Z., Ping, P., and Li, S. (2015) *A Hybrid Optimization Algorithm for Pairwise Test Suite Generation.* International Conference on Information and Automation Lijiang, China, August 2015.

Rabbi, K., Islam, R., Quazi, M., and Kaosar, M. G. (2014). *MTTG: An Efficient Technique for Test Data Generation*. 8th International Conference on Software, Knowledge, Information Management and Applications (SIKMA).

Rabbi, K., Mamun, Q. and MD Rafiqul Islam MD. (2015) *An Efficient Particle Swarm Intelligence Based Strategy to Generate Optimum Test Data in T-way Testing*. IEEE.

Rahman, M., Othman, R. R., Ahmad, R. B., and Rahman, M. M. (2015) *Event Driven Input Sequence T-way Test Strategy Using Simulated Annealing*. 5th International Conference on Intelligent Systems, Modelling and Simulation. 2015

Rahnamoun, R. (2013). *Distributed Black-Box Software Testing Using Negative Selection.* International Journal of Smart Electric Engineering 2 (2013), Pp. 151-157

Sabharwal, S., Aggorwal, M. (2016*). A Novel Approach for Deriving Interactions for Combinatorial Testing.* Engineering Science and Technology.

Sabharwal, S., and Aggarwal M. (2015) *Test Set Generation for Pairwise Testing Using Genetic Algorithms.* Journal of Information Processing Systems.

Sabharwal, S., and Aggarwal M. (2015). *Variable Strength Interaction Test Set Generation Using Multi Objective Genetic Algorithms.* International Conference on Advances in Computing, Communications and Informatics (ICACCI).

Satish, P., and Rangarajan, K. (2016) *A Preliminary Survey of Combinatorial Test Design Modeling Methods.* International Journal Of Scientific & Engineering Research, Volume 7, Issue 7, July-2016. Pp 1455-1459

Satish, P., Sheeba, K. and Rangarajan, K. (2013) *Deriving combinatorial test design model from UML activity diagram.* In Software Testing, Verification and Validation Workshops (ICSTW), 2013 IEEE Sixth International Conference on, pp. 331-337. IEEE.

Satish, P., Paul, A., Rangarajan, K. (2014). *Extracting the Combinatorial Test Parameters and Values from UML Sequence Diagrams*. IEEE International Conference on Software Testing, Verification, and Validation Workshops. Pp. 88-97

Sawant, A. A., Bari, P. H. and Chawan, P. M. (2012) *Software Testing Techniques and Strategies.* International Journal of Engineering Research and Applications (IJERA). Vol. 2, Issue 3, May-Jun 2012, Pp. 980-986

Shiba, T., Tsuchiya, T., and Kikuno, T. (2004). *Using Artificial Life Techniques to Generate Test Cases for Combinatorial Testing.* in 28th Annual International Computer Software Applications Conference, vol. 1, Hong Kong, pp 72-77.

Srikanth, A., Kulkarni, N. J., Naveen, K. V., Singh, P. and Srivastava, P. R. (2011). *Test Case Optimization Using Artificial Bee Colony Algorithm.* A. Abraham et al. (Eds.): ACC 2011, Part III, CCIS 192, Pp. 570-579. Springer-Verlag Berlin Heidelberg.

Sun, J. Z., and Wang, S. Y. (2012). *Generation of Pairwise Testing Test Sets using Novel DPSO Algorithm.* Y. Yang and M. Ma(eds.) Green Communications and Networks. LNEE, Vol. 113, Pp. 479-487. Springer, Heidelberg.

Syed Roohullah Jan, Syed Tauhid Ullah Shah, Zia Ullah Johar, Yasin Shah, and Fazlullah Khan (2016). *An Innovative Approach to Investigate Various Software Testing Techniques and Strategies.* IJSRSET. Vol 2, Issue 2, March-April 2016. Pp. 682-689.

Taylor, D. W. and Corne, D. W. (2003). *An Investigation Into Negative Selection Algorithm for Fault Detection In Refrigeration Systems.* In: Artificial Immune Systems: Proceedings of ICARIS 2003, Springer, 2003, Pp. 34-45.

Trivedi, A. H. (2012). *Software Testing Techniques.* International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE). Vol 2, Issue 10, October. Pp. 433-439

Udai, S. (2014) *A Literature Survey on Combinatorial Testing*. International Journal of Advanced Research in Computer Science and Software Engineering. Volume 4, Issue 4, April 2014, Pp 932-936. IJARCSSE

Wang, H., Gao, X. Z., Huang, X. and Song, Z. (2009). *PSO-Oprimized Negative Selection Algorithm for Anomaly Detection.* E. Avineri et al. (Eds.): Applications of Soft Computing, ASC 52, Pp. 13-21. Springer-Verlag Berlin Heidelberg.

Whittaker, J.A. (2000) *What is software testing? And why is it so hard?* Software, IEEE. 2000. 17 (1): 70-79.

Williams, A. W., and Probert R. L. (2000). *A practical strategy for testing pair-wise coverage of network interfaces.* Proceedings of the 7th International Symposium on Software Reliability Engineering, San Jose, CA, 2000. IEEE Computer Society Press:Piscataway, NJ, 2000; 246–254.

Wu, H., and Nie, C. (2014) *An Overview of Search Based Combinatorial Testing.* in Proceedings of the 7th International Workshop on Search-Based Software testing, SBST 2014, New York, NY, USA, 2014, ACm, Pp. 27-30

Xiang, L. Y., Alsewari, A. R. A. and Zamli, K. Z. (2015) *Pairwise Test Suite Generator Tool Based On Harmony Search Algorithm (HS-PTSGT).* International Journal on Artificial Intelligence, Vol. 2, Feb 2015.

Yamada, A., Biere, A., Artho, C., Kitamura, T. and Choi, E. H., (2016). *Greedy Combinatorial Test Case Generation using Unsatisfiable Cores.* International Conference on Automated Software Engineering.

Yamada, A., Kitamura, T., Artho, C., Choi E. H. and Oiwa Y. (2015). *Optimization of Combinatorial Testing by Incremental SAT Solving.* 8th IEEE International Conference on Software Testing, Verification and Validation, ICST 2015, Graz, Austria, April 13-17, 2015. Pp. 1-10.

Yang, S., Man, T. and Xu, J. (2014). *Improve Ant Algorithms for Software Testing Cases Generation.* The Scientific World Journal Vol. 2014, Article ID 392309

Yu, L., Lei, Y., Nourozborazjany, M., Kacker R. and Kuhn, D. R. (2013) *An Efficient Algorithm for Constraint Handling in Combinatorial Test Generation.* IEEE Sixth International Conference on Software Testing, Verification and Validation. 2013. USA. Pp 242-251

Zakaria, H. L., and Zamli, K. Z (2015). *Migrating Birds Optimization Based Strategies for Pairwise Testing.* 9th Malaysian Software Engineering Conference, Dec 2015.

Zhang, Z., Yan, J., Zhao, Y., and Zhang, J. (2014). *Generating Combinatorial Test Suite Using Combinatorial Optimization.* The Journal of Systems and Software. Pp. 191-207