New Strategies for Automated Random Testing

Mian Asbat Ahmad

Department of Computer Science

The University of York

A thesis submitted for the degree of

Doctor of Philosophy

April 14, 2013

Abstract

This is where you write your abstract ...

Contents

Contents ii

List of Figures iii

Nomenclature iii

1 Introduction 1

1.1 The Problems 1

1.2 Our Goals 2

1.3 Contributions 3

1.3.1 Dirt Spot Sweeping Random Strategy 3

1.3.2 Automated Discovery of Failure Domain 4

1.3.3 Invariant Guided Random+ Strategy 4

1.4 Thesis Outline 4

Appdx A 6

Appdx B 7

References 8

List of Figures

.

Acknowledgements

Several people have contributed to the completion of my PhD dissertation. However, the most prominent personality deserving due recognition is my worthy supervisor, Dr. Manuel Oriol. Thank you Manuel for your endless help, valuable guidance, constant encouragement, precious advice, sincere and affectionate attitude.

I thank my assessor Prof. John Clark for his constructive feedback on my various reports and presentations. I am also thankful and highly indebted to Prof. Richard Paige for his generous help, cooperation and guidance during my research at the University of York.

Special thanks to my father Prof. Mushtaq A. Mian who provided a conducive environment, valuable guidance and crucial support at all levels of my educational career and my very beloved mother whose love, affection and prayers have been my most precious assets. Also I am thankful to my elder brothers Dr. Ashfaq, Dr. Aftab, Dr. Ishaq, Dr. Afaq and my sister Dr. Haleema who have been the source of inspiration for me to pursue higher studies. My immediate younger brother Dr. Ilyas and my younger sister Ayesha studying in the UK, deserve recognition for their help, well wishes and moral support. Last but not the least I am very thankful to my dear wife Dr. Munazza for her company, help and cooperation throughout my stay at York.

I was funded by Departmental Overseas Research Scholarship (DORS), a financial support awarded to overseas students on the basis of outstanding academic ability and research potential. I am truly grateful to the Department of Computer Science for financial support that allowed me to concentrate on my research.

I feel it a great honour to dedicate my PhD thesis to my beloved parents for their significant contribution in achieving the goal of academic excellence.

Chapter 1

Introduction

In this chapter we give a brief introduction and motivation for the research work presented in this thesis. We commence by introducing the problems in random testing. We then describe the alternative approaches to overcome these problems, followed by our research goals and contributions. At the end of the chapter, we give an outline of the thesis.

1.1 The Problems

In software testing, one is often confronted with the problem of selecting a test data set, from a large or often infinite domain, as exhaustive testing is not always applicable. Test data set is a subset of domain carefully selected to test the given software. Finding an adequate test data set is a crucial process in any testing technique as it aims to represent the whole domain and evaluate the given system under test (SUT) for structural or functional properties [10], [8]. Manual test data set generation is a time-consuming and laborious exercise [9], therefore, automated test data set generation is always preferred. Test data generators are classified in to Pathwise, Goal-Oriented, Intellighent and Random [15]. Random test data generation generate test data set randomly from the whole domain. Unlike other approaches Random approach is simple, widely applicable, easy to implement in an automatic testing tool, fastest in computation, no overhead in choosing inputs and free from bias [5].

Despite the benefits random testing offers, its simplistic and non-systematic nature expose it to high criticism [14]. Myers & Sandler [11] mentioned it as "Probably the poorest methodology of all is random-input testing...". Where this statement is based on intuition and lacks any experimental evidence, it motivated the interest of research community to evaluate and improve random testing. Adaptive random testing [2], Restricted Random Testing [1], Feedback directed Random Test Generation [13], Mirror Adaptive Random Testing [3] and Quasi Random Testing [4] are few of the enhanced random testing techniques aiming to increase its fault finding ability.

Random testing is also considered weak in providing high code coverage [12], [6]. For example, in random testing when the conditional statement "if (x == 25) then ... " is exposed to execution then there is only one chance, of the "then..." part of the statement, to be executed out of 2^{32}. If x is an integer variable of 32 bit value [7].

Random testing is no exception when it comes to the complexity of understanding and evaluating test results. Modern testing techniques simplifies results by truncating the lengthy log files and display only the fault revealing test cases in the form of unit tests. However efforts are required to show the test results in more compact and user friendly way.

1.2 Our Goals

The overall goal of this thesis is to develop new techniques for automated testing based on random strategy that addresses the above mentioned problems. Particularly,

1. We aim to develop an automated random testing technique which is able to generate more fault-revealing test data. To achieve this we exploit the presence of fault clusters found in the form of block and strip fault domains inside the input domain of a given SUT. Thus we are able to find equal number of faults in fewer number of test cases than other random strategies.

2. We aim to develop a novel framework for finding the faults, their domains and the presentation of obtained results on a graphical chart inside the specified lower and upper bound. It considers the correlations of the fault and fault domain. It also gives a simplified and user friendly report to easily identify the faulty regions across the whole domain.

3. We aim to develop another automated testing technique which aims to increase code coverage and generation of more fault-revealing data. To achieve this we utilises Daikon— an automated invariant detector that reports likely program invariant. An invariant is a property that holds at certain point or points in a program. With these invariants in hand we can restrict the random strategy to generate values around these critical points. Thus we are able to increase the code coverage and quick identification of faults.

1.3 Contributions

To achieve the research goals described in Section xx, we make the following specific contributions:

1.3.1 Dirt Spot Sweeping Random Strategy

Development of a new, enhanced and improved form of automated random testing: the Dirt Spot Sweeping Random (DSSR) strategy. This strategy is based on the assumption that faults and unique failures reside in contiguous blocks and stripes. The DSSR strategy starts as a regular random+ testing strategy a random testing technique with preference for boundary values. When a failure is found, it increases the chances of using neighbouring values of the failure in subsequent tests, thus slowly sweeping values around the failures found in hope of finding failures of different kind in its vicinity. The DSSR strategy is implemented in the YETI random testing tool. It is evaluated against random (R) and random+ (R+) strategies by testing 60 classes (35,785 line of code) with one million (105) calls for each session, 30 times for each strategy. The results indicate that for 31 classes, all three strategies find the same unique failures. We analysed the 29 remaining classes using t-tests and found that for 7 classes DSSR is significantly better than both R+ and R, for 8 classes it performs similarly to R+ and is significantly better than R, and for 2 classes it performs similarly to random and is better than R+. In all other cases, DSSR, R+ and R do not perform significantly differently. Numerically, the DSSR strategy finds 43 more unique failures than R and 12 more unique failures than R+.

1.3.2 Automated Discovery of Failure Domain

There are several automated random strategies of software testing based on the presence of point, block and strip fault domains inside the whole input domain. As yet no particular, fully automated test strategy has been developed for the discovery and evaluation of the fault domains. We therefore have developed Automated Discovery of Failure Domain, a new random test strategy that finds the faults and the fault domains in a given system under test. It further provides visualisation of the identified pass and fail domain. In this paper we describe ADFD strategy, its implementation in YETI and illustrate its working with the help of an example. We report on experiments in which we tested error seeded one and two-dimensional numerical programs. Our experimental results show that for each SUT, ADFD strategy successfully performs identification of faults, fault domains and their representation on graphical chart.

1.3.3 Invariant Guided Random+ Strategy

Acknowledgement of random testing being simple in implementation, quick in test case generation and free from any bias, motivated research community to do more for increase in performance, particularly, in code coverage and fault-finding ability. One such effort is Random+ — Ordinary random testing technique with addition of interesting values (border values) of high preference. We took a step further and developed Invariant Guided Random+ Strategy (IGRS). IGRS is an extended form of Random+ strategy guided by software invariants. Invariants from the given software under test are collected by Daikon— an automated invariant detector that reports likely invariant, prior to testing and added to the list of interesting values with high preference. The strategy generate more values around these critical program values. Experimental result shows that IGRS not only increase the code coverage but also find some subtle errors that pure Random and Random+ were either unable or may take a long time to find.

1.4 Thesis Outline

The rest of the thesis is organised as follows: In Chapter 2, we give a thorough review of the relevant literature. We commence by discussing a brief introduction of software testing and shed light on various techniques and types of software testing. Then, we extend our attention to automated random testing and the testing tools using random technique to test softwares. In Chapter 3, we present our first automated random strategy Dirt Spot Sweeping Random (DSSR) strategy based on sweeping faults from the clusters in the input domain. Chapter 4 describes our second automated random strategy which focus on dynamically finding the fault with their domains and its graphical representation. Chapter 5 presents the third strategy that focus on quick identification of faults and increase in coverage with the help of literals; Finally, in Chapter 7, we summarise the contributions of this thesis, discuss the weaknesses in the work, and suggest avenues for future work.

Appdx A

and here I put a bit of postamble ...

Appdx B

and here I put some more postamble ...

References

[1] Kwok Ping Chan, Tsong Yueh Chen, and Dave Towey. Restricted random testing. In *Proceedings of the 7th International Conference on Software Quality*, ECSQ '02, pages 321–330, London, UK, UK, 2002. Springer-Verlag. 2

[2] T. Y. Chen. Adaptive random testing. *Eighth International Conference on Qualify Software*, 0:443, 2008. 2

[3] T. Y. Chen, F. C. Kuo, R. G. Merkel, and S. P. Ng. Mirror adaptive random testing. In *Proceedings of the Third International Conference on Quality Software*, QSIC '03, page 4, Washington, DC, USA, 2003. IEEE Computer Society. 2

[4] Tsong Yueh Chen and Robert Merkel. Quasi-random testing. In *Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering*, ASE '05, pages 309–312, New York, NY, USA, 2005. ACM. 2

[5] Ilinca Ciupa, Andreas Leitner, Manuel Oriol, and Bertrand Meyer. Experimental assessment of random testing for object-oriented software. In *Proceedings of the 2007 international symposium on Software testing and analysis*, ISSTA '07, pages 84–94, New York, NY, USA, 2007. ACM. 1

[6] David M. Cohen, Siddhartha R. Dalal, Michael L. Fredman, and Gardner C. Patton. The aetg system: An approach to testing based on combinatorial design. *Software Engineering, IEEE Transactions on*, 23(7):437–444, 1997. 2

[7] Patrice Godefroid, Nils Klarlund, and Koushik Sen. Dart: directed automated random testing. In *ACM Sigplan Notices*, volume 40, pages 213–223. ACM, 2005. 2

[8] William E Howden. A functional approach to program testing and analysis. *Software Engineering, IEEE Transactions on*, (10):997–1005, 1986. 1

[9] Bogdan Korel. Automated software test data generation. *Software Engineering, IEEE Transactions on*, 16(8):870–879, 1990. 1

[10] Thomas J McCabe. *Structured testing*, volume 500. IEEE Computer Society Press, 1983. 1

[11] Glenford J. Myers and Corey Sandler. *The Art of Software Testing*. John Wiley & Sons, 2004. 2

[12] A. Jefferson Offutt and J. Huffman Hayes. A semantic model of program faults. *SIGSOFT Softw. Eng. Notes*, 21(3):195–200, May 1996. 2

[13] Carlos Pacheco, Shuvendu K. Lahiri, Michael D. Ernst, and Thomas Ball. Feedback-directed random test generation. In *Proceedings of the 29th international conference on Software Engineering*, ICSE '07, pages 75–84, Washington, DC, USA, 2007. IEEE Computer Society. 2

[14] Lee J. White. Software testing and verification. *Advances in Computers*, 26(1):335–390, 1987. 2

[15] Wikipedia. Plagiarism — Wikipedia, the free encyclopedia, 20013. [Online; accessed 23-Mar-2013]. 1