Chapter 1

Introduction

In this chapter we give a brief introduction and motivation for the research work presented in this thesis. After brief motivation, 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 the structure of the thesis.

1.1 Motivation

Software is a very important component of computer system, essentially required to accomplish a given task. Some softwares are developed for use in simple day to day operations whiles others are for highly complex processes in specialized fields like research and education, business and finance, defense and security, health and medicine, science and technology, aeronautics and astronautics, commerce and industry, information and communication, environment and safety etc. The margin of error in mission-critical and safety-critical systems is so small that a minor fault can lead to huge economic losses to the organization [45]. Therefore, software companies leave no stone unturned to ensure the reliability and accuracy of the software. This dissertation is a humble contribution to the literature on the subject, with the aim to reduce the overall cost of software testing by devising new, improved and effective software testing techniques.

1.2 The Problem

Exhaustive testing of software is not always possible and the problem of selecting a test data set, from a large/infinite domain is often confronted. Test data set, as a subset of the whole domain, is carefully selected for testing the given software. Adequate test data set is a crucial factor in any testing technique because it represents the whole domain for evaluating the structural and/or functional properties [57], [44]. Generating test data set manually is a time-consuming and laborious exercise [49]. Therefore, automated test data set generation is always preferred. Data generators can be of different types i.e., Path-wise, Goal-Oriented, Intelligent or Random [89]. Random generator produces test data set randomly from the whole domain. Unlike other approaches random technique is simple, widely applicable, easy to implement, faster in computation, free from bias and costs minimum overhead [25]. According to --------, “Random testing is a simple and well-known technique which can be remarkably effective in discovering software bugs” [DART].

Despite the benefits of random testing, its simplistic and non-systematic nature exposes it to high criticism [88]. Myers & Sandler [59] mentioned it as, “Probably the poorest methodology of all random-input testing...”. However, -------- consider that the above stated statement is based on intuition and lacks any experimental evidence [---]. The criticism motivated the research community to look into various aspects of random testing for evaluation and possible improvement. Adaptive Random Testing (ART) [13], Restricted Random Testing (RRT) [10], Feed-back Directed Random Testing (FDRT) [75], Mirror Adaptive Random Testing (MART) [14] and Quasi Random Testing (QRT) [18] are a few of the enhanced random testing techniques.

Random testing is also considered weak in providing high code coverage [63], [30]. 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 232. If x is an integer variable of 32 bit value [40].

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 displaying only the fault revealing test cases in the form of unit tests. However, further efforts are required to show the test results of random testing in more compact and user-friendly way.

1.3 Research Goals

The goals of the research study are to develop new techniques for automated testing, based on random strategy, with the aim to achieve the following objectives:

1. To develop a new testing strategy with the capability to generate more faultfinding test data.
2. To develop a new testing technique for finding faults, fault domains and presentation of results on a graphical chart within the specified lower and upper bound.
3. To develop a testing framework with focus on increase in code coverage along with generation of more faultfinding test data.

1.4 Contributions

The main contributions of the thesis research are stated below:

1.4.1 Dirt Spot Sweeping Random Strategy

The faultfinding ability of the testing technique decreases when the failures lie in contiguous locations across the input domain. To overcome the problem, a new automated technique: Dirt Spot Sweeping Random (DSSR) strategy was developed. It is based on the assumption that unique failures reside in contiguous blocks and stripes. When a failure is identified, the DSSR strategy selects neighboring values for the subsequent tests. Resultantly, selected values sweep around the failure, leading to the discovery of new failures in the vicinity. Results indicated higher faultfinding ability of DSSR strategy as compared with R and R+ strategies.

1.4.2 Automated Discovery of Failure Domain

The existing random strategies of software testing discover the faults in the SUT but lack the capability of locating the fault domains. In the current research study, a fully automated testing strategy named, “Automated Discovery of Failure Domain (ADFD)” was developed with the ability to find the faults as well as the fault domains in a given SUT and provides visualization of the identified pass and fail domains in the form of a chart. The study includes description of the ADFD strategy, its implementation in YETI. The strategy is practically illustrated by executing several programs of one and two dimensions. The experimental results proved that ADFD strategy automatically performed identification of faults and fault domains along with graphical representation in the form of chart.

1.4.3 Invariant Guided Random+ Strategy

Another random test strategy named, “Invariant guided Random+ Strategy (IGRS)” was developed in the current research study. IGRS is an extended form of Random+ strategy guided by software invariants. Invariants from the given SUT are collected by Daikon— an automated invariant detector for reporting likely invariants and adding them to the SUT as assertions. The IGRS is implemented in YETI and generates values in compliance with the added assertions. Experimental results indicated improved features of IGRS in terms of higher code coverage and identification of subtle errors that R, R+ and DSSR strategies were either unable to accomplish or required larger duration.

1.5 Structure of the Thesis

The rest of the thesis is organized as follows: In Chapter 2, a thorough review of the relevant literature is given. It includes a brief introduction followed types of software testing followed by and the techniques followed by automated random testing tools. Chapter 3 describes Dirt Spot Sweeping Random (DSSR) strategy, which is based on sweeping of fault clusters in the input domain. Chapter 4 presents the newly developed Automated Discovery of Fault Domains (ADFD) strategy, which focuses on dynamically finding the faults and domains along with their graphical representation. Chapter 5 presents the new strategy developed with the focus on quick identification of faults and increase in code coverage with the help of assertions; Finally Chapter 7 summarizes contributions of the thesis research, discusses the strength and weaknesses of the study, gives conclusion and suggests avenues for future work.