Chapter 9

Automatic Test Sequence Generation for State Transition Testing via Ant Colony Optimization

Praveen Ranjan Srivastava
Birla Institute of Technology and Science (BITS), India

Baby
Birla Institute of Technology and Science (BITS), India

ABSTRACT

Software testing is a key part of software development life cycle. Due to time, cost and other circumstances, exhaustive testing is not feasible, that’s why there is need to automate the testing process. Generation of the automated and effective test suite is a very difficult task in the software testing process. Effective test suite can decrease the overall cost of testing as well as increase the probability of finding defects in software systems. Testing effectiveness can be achieved by the State Transition Testing which is commonly used in, real time, embedded and web-based kind of software system. State transition testing focuses upon the testing of transitions from one state of an object to other states. The tester’s main job is to test all the possible transitions in the system. This chapter proposed an Ant Colony Optimization technique for automated and fully coverage state-transitions in the system. Through proposed algorithm all the transitions are easily traversed at least once in the test-sequence.

INTRODUCTION

Software development markets are highly competitive; therefore it is a critical task to provide the quality software product to the customer within the limited time period. Software testing is the key by using which tester/organization can gain the confidence of the customer. A primary purpose of testing is to detect software errors so that defects may be discovered and corrected (Aditya p Mathur, 2007). Due to the time and cost it is not possible to test the software manually and fix the defects (Somerville Ian, 2008). Thus the use of test automation plays very important role in the software testing process (Pressman, 2005). Nowadays Artificial Intelligence
(AI) methodologies are changing the nature of test automation process (Phil McMinn, 2004). Academicians and researchers are using AI approach in the area of software testing for better accuracy. The application of AI techniques in Software Engineering (SE) is an emerging area of research (Briand, L. C, 2002, and Pedrycz, W., & Peters, J. F, 1998). In AI approach, researchers are using Ant Colony Optimization (ACO), Genetic Algorithm (GA), Tabu search (TB) techniques etc (Phil McMinn, 2004, Mark Harman 2007). These terminologies are known as metaheuristic approach (Praveen Ranjan Srivastava, 2008) and these approaches are being used in various processes of the software testing like test sequence generation, automation of testing etc. (Huaizhong LI and C. Peng LAM, 2005, & Mark Harman, 2007).

The quality of test data that we evaluate is based on the number of coverage for behavior of any software. The coverage of test data can be defined by many criteria. Testing of states is not sufficient (Offutt and Abdurazik, 1999 & Sabri A. Mahmoud, 2005), particularly for those systems which have large number of states (mainly real time system). Minimum acceptable strategy of software testing process for real time system is all-states, all-events, all-actions and all-transitions exercised at least once in the test suite (Offutt and Abdurazik, 1999 & Sabri A. Mahmoud, 2005, and Aditya P Mathur 2007). Literature survey reveals that under state transition testing process, tester focus upon the states, actions, events and transitions rather than states only (Sabri A. Mahmoud, 2005). UML State-transition diagrams are good tool to represent the state of the software under test (OMG, 2003). This chapter purposes an algorithm which uses an ACO optimization technique to generate the automatic state –transition test sequence, which is a strong level of software coverage.

An ACO algorithm (Yuan Zhan and John A. Clark, 2006) is a probabilistic technique for solving computational problems which can be used to find good paths through graph. It is inspired by the behavior of ants in finding paths from their colony to food (Marco Dorigoa and Thomas Stutzle, 2005). Huaizhong LI and C. Peng Lam, 2005 uses ACO to generate test sequences for state-based software testing. They also tried to generate test data on the basis of state coverage. Because the present market is highly competitive, it is the need of software organizations to provide good quality software product to the customer within the estimated budget, here a strong level of coverage testing technique is essential.

This chapter uses UML State-transition diagrams (OMG, 2003) for generating test suite. An UML State-transition diagram describe all of the states that an object can have, the events under which an object changes state (transitions), the conditions that must be fulfilled before the transition will occur (guards), and the activities undertaken during the life of an object (actions). State-transition diagrams are very useful for describing the behavior of individual objects over the full set of use cases that affect those objects (Lee Copeland, 2004).

The benefits of the proposed approach are: (1) manual generated test data is not always reliable. On the other hand automatic test data is reliable, because human beings are the most dynamic and error introducing entity. (2) State-transition based testing provides all state coverage. (3) It provides all event coverage. (4) It provides all transition coverage and (5) UML state-transition diagram created, which shows the static behavior of the software.

**BACKGROUND WORK**

There are three main activities associated with software testing: (1) Optimal test data generation (Biswal, B. N, 2008), (2) Execution of those optimal set of test data and (3) Analysis of the of test results. (Huaizhong LI and C. Peng LAM, 2005), The key question addressed in the area of software testing is how to select optimal test cases with high degree of defects. Since we know