

An On-Line Testing Approach Using Code-Perturbation

Zan Yang and Gwan Choi
Texas A&M University, College Station, Texas, USA

Abstract

This paper presents an efficient on-line testing approach that uses application programs as the test inputs. The approach perturbs the program-control-flow of the application code to generate test that exhausts all branching possibilities of the application code. The resulting test vector is a set of code-segments that can be sequentially issued to stimulate hardware and software functions spanned by the target system. The main impact of the approach is that the service routines that would be invoked in rare events, e.g. interrupt handling, fault detection and recovery, unusual branching, are now covered without the use of fault injection. The approach also takes a significantly less time than that the straightforward run of the code.

1. Introduction

On-line testing becomes increasingly difficult to manage as the system complexity grows. The difficulties associated with on-line testing amplify in system applications, such as the systems used in space environment, where maintenance access is physically limited. Systems that require continuous services further restrict the resources to limit the instrumentation necessary to conduct a run-time testing. Hence, on-line testing is often limited to simple routines and a low-quality test is often expected.

Unlike the rigorous manufacturing or developmental testing, the purpose of on-line field testing is to ensure the correct operational functionality of the entire system, which includes both the hardware and the software. This is apart from the objectives in manufacturing testing where the goal is to cover as exhaustively as a resource will permit the legal hardware state spaces spanned by the hardware design. Software testing and SW/HW co-testing also aim to exhaust the valid system state space in an exhaustive fashion. However, during an on-line testing it is often sufficient to cover merely the functions that are required to ensure correct and/or continuous operation of the system. Hence the issue now is to run the application programs thoroughly so that all aspects of the functionality spanned by the programs

is tested. However, this is too expensive and impractical. In practice on-line testing is conducted using codes that are often too abstract, though may be easy to execute. The objective is high-level functional coverage of the target system. A test should trail the application programs so that the testing results can ensure the correctness of the hardware when these programs are running on top of it. It is not a concern that the hardware is correct at large.

2. Test Generation Using Code-Perturbation

In this study we present a method to devise a set of testing based on the application programs that is already in place. It mimics the typical execution of the target programs so the test coverage with respect to a realistic operation of the given hardware and software may be increased. The approach makes available partial hardware resources for test, configures the system for on-line testing, generates and apply test code, and detect and diagnose the fault symptoms exhibited by the system under test. The modification include disconnecting the program counter from the instruction cache. The program execution flow is no longer solely decided by the program. The main emphasis of this paper is on the approach to accelerate the test process by generating test using code-perturbation method similar to that presented in [1] for verification. The impact of code-perturbation for on-line testing is a delivery of much more efficient test tailored to fully cover the functions that are spanned by the application program.

The test generation method is based on the static white-box code-perturbation approach. Unlike hardware verification, the completeness problem does not need to be addressed in on-line testing due to the specific focus. Two small programs are used as the examples in this study. It takes only 7.5% of the straightforward execution time to finish the testing vectors.

References

- [1] Z. Yang, B. Min and G. Choi, "Simulation Using Code-Perturbation: Black- and White-Box Approach", Proceedings of ISQED 2001