Integration of multithreaded software testing into standard engineering process

Abstract

In the multicore era, in which we are currently living, software consists of dependent threads that execute concurrently. Input data and the interaction between concurrent threads determine the result of the multithreaded software. Testing of concurrent software is a challenging task. It is necessary to create test cases with a range of input data, in order to test individual software threads. Further, it is necessary to test interaction between threads. However, due to the nondeterministic nature of multicore hardware (e.g. shared cache, shared memory bus); it is hard to predict order of interactions between threads, as they do not progress uniformly. Forcing different orders of interaction requires an intense work and due to the number of possible orderings, is not an efficient testing approach. Therefore, current approaches perform analysis of the execution trace in order to find concurrency bugs.

However, existing testing techniques are completely decoupled from software development. They are often executed independently and require a complex setup. Compared to the unit testing, testing of multithreaded software requires greater effort and is not standardized in a form that is a part of a regular software development process. Evolution of a multithreaded software is a great problem. If developers change only one thread, they want to test only the introduced change. However, current testing techniques require execution of the full software stack, in order to record an execution trace and find concurrency bugs. This process is time consuming on its own and contributes to the low usability of the existing approaches. Considering that multithreaded software is becoming predominant, this is a big industrial challenge.

In order to solve these challenges, we suggest integration of software development process with approaches for testing of concurrency software. The main idea of this work is to couple the testing techniques with the software execution. In existing software development processes, engineers use debug to understand the software. We aim to add an additional component to the software execution process, which is similar to debugger in its nature. The multithreaded software testing component (MSTC) executes in parallel with unit tests of the individual threads. The MSTC keeps track of executed unit tests of individual threads and creates a repository of accessed memory locations, to identify recognize an access to shared memory.
The main challenges:

- Integration of MSTC into build environment, to run the test software.
- Conceptual solution to support development of embedded software when considering changes introduced only to a number of threads.
- Integration of the MSTC with standard development IDE (e.g., Eclipse).

Benefits:

- Automatization of multithreaded software development and testing process.
- Sequential testing of multithreaded software.

Keywords : Concurrent Software, Software Development, LLVM

Requirements:

- Excellent C/C++ knowledge.
- Good Java knowledge.
- Understanding of compilers, and software build environments (e.g., Makefiles).
- Familiarity with Linux.

Understanding of LLVM is beneficial but not a must requirement.