Bugs and Debugging of Concurrent and Multicore Software

Sara Abbaspour Asadollah

Mälardalen University, Västerås, Sweden
sara.abbaspour@mdh.se

Abstract

Multicore platforms have been widely adopted in recent years and have resulted in increased development of concurrent software. However, concurrent software is still difficult to test and debug for at least three reasons: (1) concurrency bugs involve complex **interactions among multiple threads**; (2) concurrency bugs are **hard to reproduce** and (3) concurrent software has a **large interleaving space**. Current testing techniques and solutions for concurrency bugs typically focus on exposing concurrency bugs in the large interleaving space, but they often do not provide debugging information for developers (or testers) to understand the bugs.

Debugging, the process of identifying, localizing and fixing bugs, is a key activity in software development. Debugging concurrent software is significantly more challenging than debugging sequential software mainly due to the issues like non-determinism and difficulties of reproducing failures.

We investigate the first and second of the above mentioned problems in concurrent software with the aim to help developers (and testers) to better understand concurrency bugs. We propose a disjoint classification for concurrency bugs by classifying the bugs in a common structure considering relevant observable properties in order to map relevant studies with our proposed classification and explore concurrency-related bugs in real-world software.

We also provide an overview of existing research on concurrent and multicore software debugging. We apply a systematic mapping study method in order to summarize the recent publication trends and clarify current research gaps in the field. Based on the obtained results we summarize the publication trend in the field during the last decade by showing distributions of publications with respect to year, publication venues, representation of academia and industry, and active research institutes. We identify research gaps in the field based on attributes such as types of concurrency bugs, types of debugging processes, types of research and research contributions. The results of our mapping study also indicate that the current body of knowledge concerning debugging concurrent and multicore software does not report studies on many of the other types of bugs or on the debugging process.

Moreover, we investigate the bug reports from an open source software project (Apache Hadoop). Our results indicate that a relatively small share of bugs is related to concurrency issues, while the vast majority are non-concurrency bugs. Fixing time for concurrency and non-concurrency bugs is different but this difference is relatively small. In addition, concurrency bugs are considered to be slightly more severe than non-concurrency bugs.

A general conclusion from the investigations reveal that even if there are quite a number of studies on concurrent and multicore software debugging, there are still some issues that have not been sufficiently covered including *order violation*, *suspension* and *starvation*. Furthermore, our investigation shows it is still hard for developers and testers to distinguish concurrency bugs from other types of software bugs. An interesting agenda for future work would be to combine the evidence identified in the systematic mapping study with evidence from the case study to define hypotheses and theories which will form the basis for proposing new methods, process

and tools for concurrent and multicore software debugging. We think a possible future research is to propose solutions to bridge the identified gaps between the paradigms.