Software Engineering Seminar

Testing of Multi-threaded Programs

Description

Multi-threaded programs are usually error prune due to data races. However, testing of real-world concurrent programs can be both time- and space-consuming, since the exploration space can increase exponentially during execution [1]. To tackle this problem, existing approaches have been proposed to efficiently detect concurrency bugs [1, 2, 3, 4], where techniques such as bounded sampling, fuzzing, and dynamic slicing are usually involved.

The student should examine and discuss the current state of the art approaches for testing multithreaded programs.

References

[1] Dongjie Chen, Yanyan Jiang, Chang Xu, Xiaoxing Ma, and Jian Lu. Testing multithreaded programs via thread speed control. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2018, pages 15–25, New York, NY, USA, 2018. ACM.

[2] Baris Kasikci, Weidong Cui, Xinyang Ge, and Ben Niu. Lazy diagnosis of in-production concurrency bugs. In *Proceedings of the 26th Symposium on Operating Systems Principles*, SOSP '17, pages 582–598, New York, NY, USA, 2017. ACM.

[3] Koushik Sen. Race directed random testing of concurrent programs. *SIGPLAN Not.*, 43(6):11–21, June 2008.

[4] Long Zheng, Xiaofei Liao, Hai Jin, Bingsheng He, Jingling Xue, and Haikun Liu. Towards concurrency race debugging: An integrated approach for constraint solving and dynamic slicing. In *Proceedings of the 27th International Conference on Parallel Architectures and Compilation Techniques*, PACT '18, pages 26:1–26:13, New York, NY, USA, 2018. ACM.

Contacts

Minxing Tang (`tanminxi@informatik.hu-berlin.de`)
Software Engineering Group
Institut für Informatik
Humboldt-Universität zu Berlin