Software Engineering Seminar

Automated Software Aging Analysis

Description

Memory leaks that cause software aging are latent defects triggering memory depletion during runtime. They are usually caused by forgetting to free allocated heap space (C/C++) or to remove all references to obsolete objects (Java). Even if software is known to contain memory leaks identifying their root causes (so-called defect localization) can be tricky. Causes for this are manifold: leaks manifest slowly (i.e. they need prolonged execution time to surface); data structures like caches can keep allocated memory for legitimate reasons; it requires a variety of data to decide whether an object/memory fragment is obsolete or needed. Several approaches and tools for localization of memory leaks exist [1, 2, 3, 4, 5]. While some suffer significant performance issues [1], other require sophisticated static analysis [3] or are proprietary (e.g. LeakBot from IBM). A simpler approach [2, 4, 5] using dynamic analysis exploits regression testing and statistical analysis.

The goal of this seminar topic is to compare and evaluate several approaches for localization of memory leaks based on runtime analysis, and attempt to improve their performance and possibly also accuracy. This requires reproduction of the original research and application to a benchmark set. The work might be optionally extended for a MS Thesis if novel feature selection algorithms can be engineered.

Prerequisites

A basic knowledge of Software Engineering I/II and Requirements Engineering and Software Architectures.

References

[1] James A. Clause and Alessandro Orso. LEAKPOINT: pinpointing the causes of memory leaks. In Jeff Kramer, Judith Bishop, Premkumar T. Devanbu, and Sebastián Uchitel, editors, *Proceedings of the 32nd ACM/IEEE Int. Conf. on Software Engineering - Volume 1, ICSE 2010*, pages 515–524. ACM, 2010.

[2] Domenico Cotroneo, Francesco Fucci, Antonio Ken Iannillo, and Roberto Natellaand Roberto Pietrantuono. Software aging analysis of the android mobile os. In *Proceedings of the 2016 Int. Symp. on Software Reliability Engineering - ISSRE, 2016*, page to appear, 2016.

[3] Matthias Hauswirth and Trishul M. Chilimbi. Low-overhead memory leak detection using adaptive statistical profiling. In Shubu Mukherjee and Kathryn S. McKinley, editors, *Proc. of the 11th Int. Conf. on Architectural Support for Prog. Lang. and Oper. Systems, ASPLOS , 2004*, pages 156–164. ACM, 2004.

[4] Felix Langner and Artur Andrzejak. Detection and root cause analysis of memory-related software aging defects by automated tests. In *2013 IEEE 21st Int. Symp. on Modelling, Analysis and Simulation of Computer and Telecommunication Systems,, 2013*, pages 365–369. IEEE Computer Society, 2013.

[5] Rivalino Matias, Artur Andrzejak, Fumio Machida, Diego Elias, and Kishor S. Trivedi. A systematic differential analysis for fast and robust detection of software aging. In *33rd IEEE Int. Symp. on Reliable Distributed Systems, SRDS 2014*, pages 311–320. IEEE Computer Society, 2014.

Contacts

Lars Grunske (`grunske@informatik.hu-berlin.de`)
Software Engineering Group
Institut für Informatik
Humboldt-Universität zu Berlin