Institut für Systemsoftware
O.Univ.Prof. Dr. Hanspeter Mössenböck

JOHANNES KEPLER
UNIVERSITÄT LINZ
Netzwerk für Forschung, Lehre und Praxis

Debugging of High-Level Concurrent Applications on top of Truffle

Master thesis for …
Matr.Nr.: …
Email: …@...

With the omnipresence of multicore processors, applications have started to use concurrency abstractions in widely different, but also more and more complex ways [4]. Often applications combine multiple abstractions to solve problems and reach the required performance goals. Unfortunately, todays debugging tools rarely know more about concurrency than the basic notion of threads and locks. More high-level abstractions such as Actors [1,2], Futures [2], Software Transactional Memory (STM) [3], and others are typically not understood, which means that developers debug on the lowest system level instead being presented with the abstractions they use to build their applications.

In this thesis, we want to explore better debugging support for these high-level abstractions. One of the main problems that needs to be overcome is overhead of instrumentation to avoid that it interferes and distorts the concurrent execution of an application. Often debugging concurrent systems is hard, because as soon as the debugger is attached, race conditions disappear. In the context of this thesis, the goal is to find the minimal necessary instrumentation to reliably debug for instance an application that uses actor-like message passing, asynchronous futures, or STM.

To be able to focus on the research question, the thesis will use a language implemented in Truffle [5,6], a framework for implementing languages efficiently based on simple interpreters.

The scope of this thesis is as follows:

- Build a simple debugger, e.g., for an actor system using the Truffle instrumentation framework
- Minimize the instrumentation overhead to avoid interference with program execution

Optional goals are:

- Find strategy for minimal instrumentation overhead that still provides sufficient information to identify the cause of bugs

The work's progress should be discussed with the supervisor at least every 2 weeks. Please note the guidelines of the Institute for System Software when preparing the written thesis.

Supervisor: Dr. Stefan Marr

[1] Agha, G. ACTORS: A Model of Concurrent Computation in Distributed Systems. Cambridge, MA, USA: MIT Press, 1986.

[2] Van Cutsem, T., Gonzalez Boix, E., Scholliers, C., Lombide Carreton, A., Harnie, D., Pinte, K. and De Meuter, W. "AmbientTalk: programming responsive mobile peer-to-peer applications with actors." Computer Languages, Systems & Structures 40, no. 3–4 (2014): 112-136.

[3] Shavit, N. and Touitou, D. "Software Transactional Memory." In Proc. of PODC '95: the fourteenth annual ACM symposium on Principles of distributed computing, USA, 1995.

[4] Tasharofi, S., Dinges, P. and Johnson, R. "Why Do Scala Developers Mix the Actor Model with other Concurrency Models?." In Proc. of ECOOP 2013, 302-326. Springer, 2013.

[5] http://ssw.uni-linz.ac.at/Research/Projects/JVM/Truffle.html

[6] Würthinger, T., Wöß, A., Stadler, L., Duboscq, G., Simon D. and Wimmer, C. "Self-Optimizing AST Interpreters." In Proc. of the 8th Dynamic Languages Symposium, 2012.