Deadlock checking by data race detection

Ka I Pun[1], Martin Steffen[1] and Volker Stolz[1,2]

[1] Department of Informatics, University of Oslo, Norway
[2] United Nations University—Intl. Inst. for Software Technology, Macao

Abstract

Deadlocks are a common problem in programs with lock-based concurrency and are hard to avoid or even to detect. One way for deadlock prevention is to statically analyse the program code to spot sources of potential deadlocks.

We reduce the problem of deadlock checking to race checking, another prominent concurrency-related error for which good (static) checking tools exist. The transformation uses a type and effect-based static analysis, which analyses the data flow in connection with lock handling to find out control-points which are potentially part of a deadlock. These control-points are instrumented appropriately with additional shared variables, i.e., race variables injected for the purpose of the race analysis. To avoid overly many false positives for deadlock cycles of length longer than two, the instrumentation is refined by adding "gate locks". The type and effect system, and the transformation are formally given. We prove our analysis sound using a simple, concurrent calculus with re-entrant locks. The type system allows a context-sensitive analysis.