Software Component Technology Group

Master Project

Using program slicing to improve error reporting in Boogie

Karin Freiermuth
Supervisor: Joseph Ruskiewicz
Prof. Dr. Peter Müller

May 17, 2007

Introduction. Boogie is a static program verifier. As part of the Spec# programming environment, Boogie verifies Spec# programs. First it translates the programs into its own language, BoogiePL. BoogiePL, an intermediate language, is used for program analysis and verification. Given such a program, Boogie generates verification conditions, which are then attempted to prove by a theorem prover. If no errors can be found, it is guaranteed, that the original program is correct. In case of one or more detected errors an error message is returned.

Boogie reports an error, if a verification condition can't be verified by the theorem prover. The error report states any condition, that could not be proved. However, especially in complex programs, finding the source of the error can get tedious. The failed condition does not necessarily correspond to the source of the program error.

The technique of program slicing is used to extract those parts of a program, that might affect a given condition. The goal of program slicing is to generate a slice, that reaches a reduced complexity of the program. However, program slicing does not help in many cases, especially in the presence of heap like structures, where the size of the slice may approach that of the original program.

Goal of this masters's project. Error understanding in Boogie using program slicing. In addition to program slicing we take advantage of the failed verification condition and the theorem prover. The combination of all of them may turn program slicing into an attractive technique to improve error localization in Boogie.

The main parts of this project.

1. Design of a technique of program slicing in Boogie using the error trace in combination with the theorem prover and with static analysis

2. Implementation of program slicing in Boogie
3. Evaluation and determination of the appropriate technique considering different cases

Possible extensions.

1. Improvement of error reports by not just specifying the place of the error, but the reason for its occurrence
2. Extending the error report to Spec# level