Karlsruher Institut für Technologie

Institute of Information Security and Dependability (KASTEL)
Application-oriented Formal Verification
Prof. Dr. Bernhard Beckert

Bachelor thesis

Comparing lightweight vs. heavyweight verification

Background. Pluggable type systems are a lightweight alternernative to more complex specification approaches. At our chair, we have developed *property types* for Java. A property type consists of a Java type and an annotation associated with a boolean property, e.g., `min <= subject <= max`.

```
@Interval(min=1, max=3) int f;
public void foo() {
    f = 3; // OK
    @Interval(min=2, max=4)
    int l = f + 1; // Error!
}
```

We have developed a type checker than can prove the correctness of most simple parts of the program. For the difficult-to-prove parts, it translates the properties to be proven into a format that can be used by an SMT solver, or—if this also does not suffice—the deductive verification tool KeY. But so far, there is no large-scale case study comparing this verification approach to multiple possible alternatives.

Task. Your task is to specify and verify the correctness of a medium to large program using the following approaches, and then compare the overhead in both specification and verification:

1. Our approach described here.
2. A pure type-system–based approach, where instead of relying on an SMT solver and KeY, we attempt to use more expressive type systems. E.g., the Value Checker from the Java Checker Framework is able to deal with our example program without invoking outside tools, but may not be expressive enough for some other cases.
3. Deductive verification using KeY only, or using KeY alongside its built-in SMT support.

Requirements. You have experience in Java programming and basic knowledge of logic and formal verification, e.g., from the *Formal Systems* lecture.

Contact
Florian Lanzinger lanzinger@kit.edu Office 50.34R227