Karlsruher Institut für Technologie

Institut für Theoretische Informatik (ITI)
Anwendungsorientierte Formale Verifikation
Prof. Dr. Bernhard Beckert

Bachelor/Master Thesis

Algebraische Datentypen in KeY

Hintergrund. Deduktive Programmverifikation beschäftigt sich mit dem formalen Nachweis der Korrektheit von Programmen. Was „korrekt“ für ein konkretes Programm bedeutet, wird über eine formale Spezifikation definiert und mit einem logischem Kalkül automatisch oder (in komplizierteren Fällen) benutzergeführt bewiesen.

```
spec  GENERATED_SET [sort Elem] =
      generated type Set ::= empty | insert(Elem; Set)
      pred __is_in__ : Elem × Set
      ops  {__}(e : Elem) : Set = insert(e, empty);
           __ ∪ __        : Set × Set → Set;
           remove         : Elem × Set → Set
      ∀e, e' : Elem; S, S' : Set
      • ¬(e is_in empty)
      • e is_in insert(e', S) ⇔ (e = e' ∨ e is_in S)
      • S = S' ⇔ (∀x : Elem • x is_in S ⇔ x is_in S')      %(equal_sets)%
      • e is_in (S ∪ S') ⇔ (e is_in S ∨ e is_in S')
      • e is_in remove(e', S) ⇔ (¬(e = e') ∧ e is_in S)
then %implies
      ∀e, e' : Elem; S : Set
      • insert(e, insert(e, S)) = insert(e, S)
      • insert(e, insert(e', S)) = insert(e', insert(e, S))
      generated type Set ::= empty | {__}(Elem) | __ ∪ __(Set; Set)
      op __ ∪ __ : Set × Set → Set, assoc, comm, idem, unit empty
end
```

von CASL User Manual

Häufig lässt sich dabei ein Programm einfach über Algebraische Datentypen spezifiziert. Diese mathematische Entitäten zeichnen sich durch angenehme Eigenschaften beim Beweisen aus: Sie sind unveränderlich und wohldefiniert. Allerdings sind diese bisher nur sehr aufwendig in KeY zu definieren. KeY ist ein interaktiver Theorembeweiser aus unserer Forschungsgruppe, mit dem man die Einhaltung von Java-Programmen gegenüber einer formalen Spezifikation nachweisen kann.

Aufgabe. In dieser Arbeit möchten wir die *Common Algebraic Specification Language* (CASL) für KeY zugänglich machen. Dazu bauen wir eine Übersetzung, die CASL in typisierte Prädikatenlogik und Schlussregel von KeY übersetzt. Anteil der unterstützten CASL-Konstruke richtet sich nach Art der Arbeit. Neben der praktischen Entwicklung Übersetzung muss dabei der theoretischen Hintergrund umfassend behandelt, sowie eine ansprechende Fallstudie zur Anwendbarkeit.

Ihr Profil. Sie sollten solide Programmier-Kenntnisse in Java besitzen und das Modul *Formale Systeme* sehr erfolgreich abgeschlossen haben.

Kontakt
Alexander Weigl weigl@kit.edu Office: 50.34, R224