

# Modul “Programmverifikation”

## Ziel

Einführung in die Methoden zum Nachweis der Korrektheit von sequentiellen und parallelen Programmen. Es werden folgende Kenntnisse, Fertigkeiten und Kompetenzen vermittelt:

- Kenntnisse:  
operationelle Semantik für sequentielle, parallele und verteilte Programme, partielle und totale Korrektheit, Hoaresche Beweisregeln für while-Programme, Invarianten und Terminierungsfunktionen, formale Beweisskizzen, Sätze über Korrektheit und Vollständigkeit von Beweissystemen, Beweissysteme für drei Klassen paralleler Programme (disjunkt, mit gemeinsamen Variablen, mit Synchronisation), Beweismethode von Owicki und Gries, Interferenz-Freiheit, Deadlock-Freiheit, Transformation paralleler Programme in nichtdeterministische Programme, Beweissysteme für verteilte Programme, Fairness, diverse Fallstudien
- Fertigkeiten:  
Ein-Ausgabe-Spezifikation von Programmen aufstellen, manuelle Verifikation von Programmen verschiedener Klassen mit Hilfe von Beweisregeln, Prüfen der Interferenz- und Deadlock-Freiheit paralleler Programme
- Kompetenzen:  
Verständnis operationeller Semantik von sequentiellen, parallelen und verteilten Programmen, Konzept der partiellen und totalen Programmkorrektheit verstanden, Verständnis von Korrektheit und Vollständigkeit von Beweissystemen, Transformation paralleler und verteilter Programme in nichtdeterministische Programme

## Inhalt

Programmverifikation ist ein systematischer Ansatz, die Fehlerfreiheit von Programmen zu zeigen. Dazu wird bewiesen, dass ein vorgegebenes Programm bestimmte wünschenswerte Verhaltens-Eigenschaften besitzt. Beispielsweise sollte ein Sortierprogramm nur sortierte Felder als Ergebnis abliefern.

Bei sequentiellen Programmen geht es dabei vor allem um partielle Korrektheit, Terminierung und Abwesenheit von Laufzeitfehlern. Bei parallelen Programmen sind zusätzliche Verhaltens-Eigenschaften wichtig: Interferenz-Freiheit, Deadlock-Freiheit und faires Ablaufverhalten.

In der Vorlesung geht es vornehmlich um die Verifikation paralleler Programme. Dazu werden klassische Methoden der Hoareschen Logik mit neuen Techniken der Programmtransformation kombiniert. Als Vorbereitung werden zunächst sequentielle Programme behandelt.

## Literatur

Als Grundlage für die Vorlesung dient das Buch

“K.R. Apt, E.-R. Olderog, *Programmverifikation*, Springer-Verlag, 1994”

oder auch die erweiterte englische Ausgabe

“K.R. Apt, E.-R. Olderog, *Verification of Sequential and Concurrent Programs*, Second Edition, Springer-Verlag, 1997”.