MSE | MASTER OF SCIENCE IN ENGINEERING

Forschungsprojektarbeit des Profils Computer Science

Verified Software Construction in the Age of AI

Ausgangslage

Large Language Models generieren heute einen wachsenden Anteil des produktiven Codes: Microsoft, Google und AWS berichten von 25–30% AI-generiertem Code. Was fehlt, sind Garantien. Unterspezifizierte Prompts führen zu unterspezifizierter Software, klassische Qualitätssicherung skaliert nicht mit der Geschwindigkeit der Generierung. Der Verifikations-Gap wächst.

Formale Verifikation bietet einen Ausweg. Property-based Testing ist ein erster Schritt: die Software wird gegen randomisierte Eingaben getestet, was LLMs verunmöglicht, durch fake Implementierungen gezielt einzelne Testfälle zu erfüllen. Zudem erfordert die Formulierung von Properties ein explizites Nachdenken über Eigenschaften der Software. Leans Dependent Types gehen weiter und erlauben es, Vorbedingungen, Nachbedingungen und Invarianten direkt im Typsystem maschinengeprüft zu verifizieren. Ein Effektsystem ergänzt dies auf Architekturebene: welche Komponenten dürfen auf die Datenbank zugreifen, welche HTTP-Calls machen, welche sind garantiert seiteneffektfrei? Mit Lean steht eine Plattform zur Verfügung, die Programmiersprache und Theorembeweiser vereint und von der AI-Community breit adoptiert wird.

Der generierte Code kann dabei wie Assembly betrachtet werden: er muss nicht verstanden werden, solange maschinengeprüfte Garantien sicherstellen, dass er das tut, was von ihm verlangt wird.

Zielsetzung / Methodik / Vorgehen

Das Projekt untersucht den folgenden Workflow: Der Mensch formuliert und versteht die Spezifikation, das LLM generiert den Code, ein Verifikationstool prüft die Konformität. Bei wiederholtem Fehlschlagen erfolgt manuelles Refinement. Konkret werden folgende Stufen untersucht:

1. **Einarbeitung in Lean und Property-based Testing:** Grundlegende Konzepte von Leans Typsystem und Dependent Types werden erarbeitet. Anhand einfacher Beispiele werden Eigenschaften als QuickCheck-analoge Properties formuliert und automatisch getestet. Ziel ist ein solides Verständnis der eigenschaftsorientierten Denkweise als Basis für die weiteren Stufen.
2. **Dependent Types und Effektsysteme:** Eigenschaften und Constraints werden in den Typ gehoben und maschinengeprüft verifiziert. Architekturelle Invarianten werden durch Effektannotationen ausgedrückt. Der Typchecker deckt Abweichungen des generierten Codes unmittelbar auf.
3. **Architekturspezifikation und LLM-Integration:** Ganze Architekturen werden durch Typen, Effekte und formale Constraints beschrieben. Der LLM erhält diese Spezifikation als Kontext und generiert konforme Implementierungen in einem iterativen, verifikationsgetriebenen Prozess.

Als Resultat wird ein Workflow erwartet, der demonstriert, wie LLMs und formale Verifikation synergetisch eingesetzt werden können, und der die zentrale Frage adressiert: Wer verifiziert den Code, wenn AI ihn schreibt?

Verlangte Skills: Kenntnisse einer puren funktionalen Sprache (Haskell oder Lean), Interesse an Typsystemen und formalen Methoden, Interesse an der Schnittstelle von KI-Werkzeugen und Softwareverifikation.

Teilaufgaben

Das Projekt kann in drei Stufen durchgeführt werden: P7 (Einarbeitung in Lean, Dependent Types und Property-based Testing), P8 (Dependent Types, Effektsystem-Design und Architekturspezifikation in Lean) und P9 (Architekturspezifikation und LLM-Integration mit verifikationsgetriebenem Feedback-Loop).

Studienart: [x] Vollzeitstudium
[x] Teilzeitstudium 50%

Projektorganisation: Einzelarbeit (im Umfeld eines Projektteams)

Projektfinanzierung: internal (Vorarbeit für spätere Projektanträge)

Arbeitsort: Windisch

Advisor: Daniel Kröni