Dekomposition für die kompositionelle Verifikation

Zusammenfassung

Die Softwareentwicklung im Bereich von sicherheitskritischen Systemen stellt eine große Herausforderung dar, da Systemfehler lebensgefährliche Konsequenzen haben können. Die Korrektheit von Software ist essentiell, um ihre Verlässlichkeit zu garantieren. Für den Korrektheitsbeweis eines Softwaremodells eignen sich integrierte Formalismen, welchen eine formale Semantik zu Grunde liegt. Das Model Checking von Softwaremodellen wird durch verschiedene Hindernisse erschwert. Die größte Herausforderung ist die Bewältigung der *Zustandsexplosion*, des exponentiellen Wachstums des Zustandsraums mit der Größe des betrachteten Systems.

Eine Reihe von Techniken beschäftigt sich mit diesem populären Problem, unter anderem die *kompositionelle Verifikation*. Die grundlegende Idee bei der kompositionellen Verifikation ist die Zerlegung des Korrektheitsbeweises in Teilaufgaben. Diese Methodik vermeidet die Konstruktion des Zustandsraums des gesamten Systems, stattdessen werden die Zustandsräume der einzelnen Systemkomponenten betrachtet. Die Anwendbarkeit dieser Technik ist an zwei Voraussetzungen gebunden. Zum einen muss das Softwaremodell *aus mehreren Einzelkomponenten zusammengesetzt sein*, was im Allgemeinen nicht der Fall ist. Des Weiteren muss die Anwendung der kompositionellen Verifikation einen *Effizienzvorteil* gegenüber dem direkten Model Checking erbringen.

Diese Arbeit beschäftigt sich mit der Dekomposition von Softwaremodellen, spezifiziert in dem integrierten Formalismus *CSP-OZ*. Eine solche Zerlegung definiert zwei Komponenten, welche sich für die kompositionelle Verifikation eignen. Eine erste Herausforderung dieser Arbeit stellt ein *Korrektheitsbeweis* dar, welcher die Äquivalenz der ursprünglichen Spezifikation und einer Dekomposition in der zugrunde liegenden semantischen Domäne zeigt. Dazu wird eine *Abhängigkeitsanalyse* durchgeführt, die auf dem Abhängigkeitsgraphen einer Spezifikation basiert. Diese Analyse führt zu einer Menge von *Korrektheitsbedingungen*, auf deren Basis der Graph in zwei Teile zerlegt wird. Daraus ergibt sich die Dekomposition der Spezifikation. Zusätzlich werden Techniken und Algorithmen zur Wiederherstellung des Kontroll- und Datenflusses der ursprünglichen Spezifikation vorgestellt. Eine zweite Schwierigkeit betrifft die *Praktikabilität* der kompositionellen Verifikation. Dazu werden in dieser Arbeit *Heuristiken* zur Messung der Qualität einer validen Dekomposition ermittelt, wobei ineffiziente Dekompositionen vernachlässigt werden. Dies erlaubt es, ausschließlich solche Zerlegungen zu betrachten, die eine *effektive* kompositionelle Verifikation in Aussicht stellen.

Insgesamt ermöglicht die beschriebene Technik die Anwendung von kompositioneller Verifikation, da sich der Ansatz nicht nur auf zusammengesetzte Systeme beschränkt. Außerdem sind durch die Heuristiken favorisierte valide Dekompositionen vorteilhaft für die Anwendung der kompositionellen Verifikation. Für den gesamten Ansatz existiert eine Werkzeugunterstützung. Diese basiert auf einer Integration in eine grafische Modellierungsumgebung, welche die *Modellierung*, *Analyse*, *Dekomposition* und *(kompositionelle) Verifikation* von integrierten Spezifikationen erlaubt. Das Model Checking wird im Rahmen eines Frameworks im Kontext des *Assume-Guarantee Beweisverfahrens* durchgeführt. Dabei werden zwei Beweisregeln verwendet, deren Korrektheit gezeigt wird. Schließlich werden einige Fallstudien sowie experimentelle Ergebnisse präsentiert.