Verifikation von Erlang Programmen mittels Abstrakter Interpretation und Model Checking

Frank Huch
Lehrstuhl für Informatik II, RWTH Aachen, 52056 Aachen, Germany
huch@i2.informatik.rwth-aachen.de

Bei der Entwicklung Verteilter Systeme ergeben sich neue, aus der sequentiellen Programmierung nicht bekannte Probleme, wie z.B. Deadlocks oder Lifelocks. Ein einfaches Testen solcher Programme reicht in der Regel nicht mehr aus, da Verteilte Systeme im Gegensatz zu sequentiellen Programmen ein nichtdeterministisches Verhalten haben und die Ausführung auf einer bestimmten Plattform mit einem bestimmten Scheduler stets nur einen konkreten Lauf des Systems bestimmt. Auf einer anderen Plattform kann sich ein völlig anderes Verhalten zeigen, insbesondere z.B. ein Deadlock.

Die Verifikation solcher Systeme während der Entwicklungsphase ist also notwendig. Bei der Verifikation von Kommunikationsprotokollen hat sich die Technik des Model Checking bewährt. Hierbei werden die Eigenschaften, welche das System haben soll, in einer Modal- oder Temporallogik spezifiziert. Bildet das Systemverhalten (z.B. beschrieben durch eine CCS-Beschreibung) eine endliche Kripke-Struktur, so kann die spezifizierte Eigenschaft verifiziert werden. Für Programmiersprachen ist dies nicht möglich, da die Semantik eines Programms in der Regel eine unendliche Struktur ist und für diese das Model Checking Problem im allgemeinen unentscheidbar ist. Deshalb versuchen wir mit Hilfe von Abstraktion dennoch Eigenschaften von nebenläufigen Programmen nachzuweisen.

Wir haben einen guten Ansatz für Erlangprogramme gefunden. Hierbei verwenden wir die operationelle Semantik von Erlang und eine Finite Domain Abstrakte Interpretation. Die Kommunikation in Erlangprogrammen erfolgt über asynchrones Message Passing, wobei gesendete Nachrichten in der Messagequeue des empfangenden Prozesses abgelegt werden. Ein Prozeß greift auf seine empfangenen Werte mittels Pattern Matching zu. Der Kontrollfluß wird also insbesondere durch bestimmte Pattern der Messages gesteuert. Deshalb werden in unserer Abstraktion aus einem gegebenen Erlangprogramm alle vorkommenden Pattern extrahiert und die Berechnungswerte auf diese eingeschränkt. Hierdurch werden aber ggf. einige Verzweigungen in der operationellen Semantik nicht mehr entscheidbar und wir erhalten zusätzlichen Nichtdeterminismus. Für eine bestimmte Teilklasse von Erlang ergibt die abstrakte operationelle Semantik ein endliches Transitionssystem mit der Eigenschaft, daß jeder Pfad der operationellen Semantik auch in der abstrahierten Semantik vorkommt. Betrachten wir Eigenschaften, die auf allen Pfaden einer Kripke-Struktur gelten sollen, wie z.B. in der Logik LTL, so ergibt sich für diese: erfüllt das abstrahierte Transitionssystem die Eigenschaft, so auch das Programm. Da die abstrakte Semantik ein endliches Transitionssystem bildet, kann die Eigenschaft mittels Model Checking überprüft werden.

In dem Vortrag wird zunächst eine formal operationelle Semantik (OS) für Erlang vorgestellt. Dann präsentiere ich einen Rahmen für Abstrakte Interpretationen über Erlangprogrammen, welcher die Eigenschaft gewährleistet, daß alle Pfade der OS auch in der Abstraktion vorhanden sind. Es wird eine konkrete Finite Domain Abstraktion vorgestellt, mit der Eigenschaften von Systemen bewiesen werden können. Diese Abstrakte Interpretation wurde bereits prototypisch implementiert und Eigenschaften wie Deadlockfreiheit, oder wechselseitiger Ausschluß wurden für einige Beispielprogramme bewiesen.