Eine typbasierte Nichtdeterminismusanalyse für logisch-funktionale Programmiersprachen*

Michael Hanus Frank Steiner
RWTH Aachen, Informatik II, D-52056 Aachen, Germany
{hanus,steiner}@informatik.rwth-aachen.de

Programmanalyse bietet vielfältige Möglichkeiten, um Verhalten und Eigenschaften von Programmen zu beschreiben. Neben der Optimierung der Programme in verschiedenen Stufen der Compilierung ist man auch an Aussagen über die Programmsicherheit und -korrektheit interessiert, beispielsweise bei der Analyse von Seiteneffekten, Kommunikationsverhalten oder Ausnahmen.

Typ- und Effektsysteme stellen eine Möglichkeit der Programmanalyse dar. Dabei wird die Sprache mit einem Typsystem versehen, das sowohl einem klassischen Typsystem funktionaler Sprachen entsprechen kann als auch einen selbstdefinierten Bereich von Typen verwenden kann, der auf die interessierenden Eigenschaften zugeschnitten wird. Zusätzlich werden Funktionen mit einem Effekt versehen, der Auswirkungen beschreibt, die bei der Auswertung der Funktion auftreten können. Ein solcher Effekt kann beispielsweise eine ausgelöste Ausnahme oder das Beschreiben einer Speicherstelle sein.

Wir wollen hier am Beispiel der Sprache Curry eine Analyse für logisch-funktionale Sprachen vorstellen, die mit Hilfe eines Typ- und Effektsystems nichtdeterministische Berechnungen erkennt, die während der Laufzeit auftreten können. Curry ist eine deklarative Sprache, die logische, funktionale und nebenläufige Paradigmen vereint. Der logische Anteil ermöglicht insbesondere den Umgang mit nichtdeterministischen Berechnungen, die beispielsweise durch Instantiierung freier Variablen mit verschiedenen Mustern ausgelöst werden können. Bei Auftreten eines solchen "don't-know"-Nichtdeterminismus wird die Berechnung aufgespalten, und die verschiedenen Zweige werden unabhängig voneinander weiterverfolgt.

Solche Aufspaltungen stellen ein Problem dar, wenn sie in einem Programm auftreten, das monadische Ein-/Ausgabe verwendet. In dem monadischen Modell werden Ein- und Ausgaben als Transformationen auf einer äußeren Welt verstanden werden, die z.B. das Dateisystem, den Bildschirm, aber auch das Internet umfaßt. Eine Vervielfältigung dieser Außenwelt ist natürlich nicht möglich, und daher müssen nichtdeterministische Berechnungen, die eine solche Vervielfältigung bewirken würden, eingekapselt werden. Andernfalls können sie Laufzeitfehler oder Programmabstürze verursachen. Um dies zu verhindern bietet Curry die eingekapselte Suche an, die dem Benutzer neben der Einkapselung nichtdeterministischer Berechnungen auch flexible Suchstrategien zur Verfügung stellt. Besonders in grösseren Programmen und bei nebenläufigen Berechnungen kann jedoch das Erkennen von möglichen nichtdeterministischen Berechnungen für den Programmierer sehr schwierig und unübersichtlich sein.

Deshalb soll mit unserer Analyse ein Werkzeug zur Verfügung gestellt werden, das den Programmierer beim Erkennen und anschließenden Einkapseln nichtdeterministischer Berechnungen unterstützt und dadurch die Programmsicherheit durch Verhinderung von Laufzeitfehlern erhöht. Eine automatische Einkapselung durch ein Tool ist ebenfalls denkbar.

Die Entwicklung der Analyse ist erst vor kurzem begonnen worden und wird sicher noch einige Zeit in Anspruch nehmen. Der Vortrag wird daher keine abgeschlossene Arbeit, sondern *work in progress* vorstellen.

*Diese Forschungsarbeit wird teilweise unterstützt von der Deutschen Forschungsgemeinschaft (DFG) (Projekt Ha 2457/1-1).