Datenfluss in bedarfsgesteuerten Berechnungen

Sebastian Fischer
Institut für Informatik
Christian-Albrechts-Universität zu Kiel

Herbert Kuchen
Institut für Wirtschaftsinformatik
Westfälische Wilhelms-Universität Münster

Quelltext-Überdeckung ist ein häufig verwendeter Maßstab zur Bewertung der Qualität von Programmtests. Sie erlaubt eine Beurteilung der durchgeführten Testläufe anhand der von ihnen ausgeführten Teile des Programms. Überdeckungs-Kriterien für imperative Sprachen reichen von einfachen Kriterien wie

- *Anweisungs-Überdeckung*, bei der überprüft wird, welche im Programm notierten Anweisungen ausgeführt werden, über
- *Kontrollfluss-Überdeckung*, bei der überprüft wird, welche Knoten bzw. Kanten des dem Programm zugeordneten Kontrollfluss-Graphen, durchlaufen werden, bis hin zu
- *Datenfluss-Überdeckung*, bei der überprüft wird, welche Zuweisungen an Variablen welche ihrer Verwendungen beeinflussen.

Jüngst gewinnt Quelltext-Überdeckung auch im Bereich funktionaler Programmierung an Bedeutung, da die Programmiersprache Haskell in zunehmendem Maße für die Entwicklung massentauglicher Anwendungen genutzt wird. Die Entwickler des Window-Managers Xmonad, verwenden Haskell Program Coverage (HPC), um die Qualität ihrer durchgeführten Tests zu bewerten. HPC verwendet ein vergleichsweise einfaches Überdeckungs-Kriterium, überprüft nämlich, welche im Programm notierten Ausdrücke von der Berechnung angefordert wurden. Diese Form der Überdeckung hat zwei wesentliche Vorzüge:

1. sie lässt sich sehr effizient aufzeichnen und
2. sie lässt sich dem Benutzer anschaulich päsentieren – sowohl in Form prozentualer Zusammenfassungen als auch durch farbig angereicherte Programmtexte, in denen markiert ist, welche Ausdrücke nicht verwendet wurden.

Gelegentlich suggeriert HPC allerdings zu viel Vertrauen in durchgeführte Tests. So gibt es Beispiele, in denen HPC 100% Überdeckung bescheinigt, Fehler im Programm jedoch nach den durchgeführten Testläufen unentdeckt bleiben. Prinzipiell kann kein Überdeckungs-Kriterium die Fehlerfreiheit von Programmen garantieren. Es lohnt sich dennoch, solche Kriterien zu entwickeln, die möglichst gründliche Tests erfordern.

Im letzten Jahr haben wir ein Werkzeug zur Testfall-Generierung für die Programmiersprache Curry vorgestellt, das eine minimale Anzahl von Testfällen anhand von Kontrollfluss-Überdeckungs-Information berechnet. Dieses Werkzeug verwendet den der Programmiersprache zu Grunde liegenden Auswertungsmechanismus *Narrowing*, der es ermöglicht, Berechnungen mit unbekannter Information durchzuführen. Testfälle werden hier dadurch erzeugt, dass die zu testende Funktion mit unbekannten Eingaben aufgerufen wird und die Eingaben während der Berechnung gebunden werden.

Wir haben unser Werkzug um ein neues Kriterium für Datenfluss-Überdeckung in deklarativen Programmen mit bedarfsgesteuerter Auswertung erweitert. Anders als in imperativen Sprachen basiert unser Kriterium nicht auf Zuweisungen an und Verwendungen von Variablen sondern auf algebraischen Datentypen und Musteranpassung. Im Wesentlichen fließen Daten von Programmstellen, an denen ein Konstruktor notiert ist, zu Programmstellen, an denen ein Muster diesen Konstruktor verwendet. Eine andere Form von Datenfluss entsteht durch Funktionen höherer Ordnung: (partiell angewendete) Funktionen *fließen* zu ihrer Applikation.

Da im Allgemeinen unentscheidbar ist, welcher Datenfluss durch ein Programm möglich ist und ferner eine Approximation ohne eine Programmausführung nicht-triviale Programm-Analysen erfordert, haben wir eine Programmtransformation entwickelt, die ein beliebiges deklaratives Programm so anreichert, dass es neben seinem eigentlichen Ergebnis auch den durch die Berechnung induzierten Datenfluss berechnet. Das transformierte Programm ist selbst rein deklarativ, verwendet also keine Seiteneffekte zur Protokollierung des Datenflusses. Auch wird die Reihenfolge der Auswertung des ursprünglichen Programms nicht verändert. Die berechnete Überdeckung entspricht also der bedarfsgesteuerten Auswertung des ursprünglichen Programms.