

Semesterarbeit

Bytecode-Unterstützung für das Universe Type System

Aliasing tritt in der objektorientierten Programmierung auf, wenn ein Objekt von mehreren anderen Objekten referenziert wird. So wird zum Beispiel eine Zelle einer doppelt verketteten Liste im Allgemeinen von ihrem Vorgänger, ihrem Nachfolger und möglicherweise vom Listenkopf sowie von Iteratoren referenziert. Während Aliasing grundsätzlich zur Flexibilität und Effizienz objektorientierter Programme beiträgt, ist es in anderen Fällen unerwünscht und Ursache von Fehlern: Die Existenz von Aliasen kann dazu führen, dass Objektstrukturen in unvorhergesehener Weise manipuliert werden, was zu Inkonsistenzen und zur Verletzung von Invarianten führen kann. Zum Beispiel geht man bei der Implementierung doppelt verketteter Listen üblicherweise davon aus, dass die Zell-Struktur nur durch den Listenkopf verändert wird. Unerwünschte Referenzen auf Zellen könnten jedoch benutzt werden, um die Struktur zu zerstören.

In der Software Component Technology Group wurde das Universe Type System zur Alias-Kontrolle entwickelt und in einem Compiler implementiert. Mit diesem Typsystem kann der Heapspeicher in sogenannten Universes strukturiert werden, wodurch man starke, statisch geprüfte Kontrolle über Referenzen zwischen Universes bekommt. Neben Vorteilen für die formale Beweisführung von Programmen ermöglicht das Typsystem die Schnittstelle von Komponenten genau zu definieren.

Aufgabenstellung dieser Semesterarbeit ist es Annotationen des Universe Type System direkt im Bytecode der Klassen zu speichern und zur Typprüfung zu verwenden. Das Problem kann in folgende Teile zerlegt werden:

1. Design eines Formats in dem die zusätzlichen Informationen in den `.class` Dateien abgespeichert werden können. Der entstehende Bytecode muss den entsprechenden Spezifikationen folgen und auf bestehenden Java Virtuellen Maschinen funktionieren.
2. Veränderung des Universe-Compilers, so dass er diese erweiterten Bytecode-Dateien erzeugt.
3. Der Compiler soll die erweiterte Typinformation bestehender Klassen bei der Übersetzung benutzen.

Aufgabensteller

Prof. Peter Müller, Werner Dietl

dietlw@inf.ethz.ch

RZ J8 / J9

ETH Zürich