JOHANNES
KEPLER
UNIVERSITÄT
LINZ

Institut für Praktische Informatik

O.Univ.-Prof. Dr. Hanspeter Mössenböck
A-4040 Linz/Auhof
Tel.: +43-732-2468 7130, Fax: +43-732-2468 7138
E-Mail: moessenboeck@ssw.uni-linz.ac.at
http://www.ssw.uni-linz.ac.at/

Registerallokation nach dem Linear-Scan-Verfahren für einen Java Just-in-Time-Compiler

Diplomaufgabe für Christian Wimmer
Matr.Nr.: 0055117

Java-Programme werden erst beim Laden in Maschinencode übersetzt (just in time compilation, JIT compilation). Dabei tritt die widersprüchliche Forderung auf, daß die Übersetzung möglichst schnell sein soll, damit man beim Laden keine Verzögerung bemerkt, daß aber andererseits der erzeugte Maschinencode möglichst gut sein soll, was ausführliche und meist zeitraubende Optimierungen erfordert.

Eine der wichtigsten Compileroptimierungen ist die Registerallokation, bei der versucht wird, lokale Variablen und temporären Werte über möglichst lange Strecken in Registern zu halten, da der Zugriff auf Register wesentlich effizienter ist als ein Speicherzugriff. Das heute am meisten verwendete Registerallokationsverfahren basiert auf Graphfärbung und ist in seiner Komplexität NP-vollständig. Obwohl heuristische Varianten dieses Verfahrens existieren, die die Laufzeitkomplexität auf $O(n^2)$ reduzieren, ist das für einen JIT-Compiler immer noch zu langsam.

Für JIT-Compiler wurde daher in den letzten Jahren das Linear-Scan-Verfahren entwickelt, das zwar nicht ganz so guten Code erzeugt, aber in der Laufzeit im wesentlichen linear ist. Ziel dieser Diplomarbeit ist es, das Linear-Scan-Verfahren für den Java HotSpot™ Client Compiler von Sun Microsystems zu implementieren.

Der Java HotSpot Compiler analysiert den zu übersetzenden Java-Bytecode und baut einen Kontrollflußgraphen mit einer Zwischensprache in Static Single Assignment Form (SSA-Form) auf. Bei der Codeerzeugung wird derzeit ein sehr einfaches Registerallokationsverfahren verwendet, das zwar schnell ist, aber Werte bei jedem Zugriff aus dem Speicher lädt. Dieser Registerallokator ist durch einen Linear-Scan-Allokator zu ersetzen, wobei folgende Probleme zu lösen sind.

- Ermitteln Sie aus dem Kontrollflußgraphen und der Zwischensprache die Live-Intervalle aller Werte im Programm. Sortieren Sie sie nach aufsteigender Anfangsposition. Dabei sind die in SSA-Form vorhandenen Phi-Funktionen zu berücksichtigen, die im Standard-Linear-Scan-Verfahren nicht beschrieben sind.
- Implementieren Sie das Linear-Scan-Verfahren, wie es in der Literatur beschrieben ist. Achten Sie darauf, daß Live-Intervalle "Löcher" aufweisen können, in denen ein Wert nicht lebt.
- Falls an einer bestimmten Stelle des Programms mehr Live-Intervalle leben als Register vorhanden sind, muß ein bereits zugewiesenes Intervall in den Speicher ausgelagert werden (spilling). Das Intervall ist an dieser Stelle zu spalten, sollte aber später wieder die Chance bekommen, in ein anderes Register geladen zu werden.
- Die Zielmaschine für Ihre Implementierung ist der Intel x86-Prozessor. Beachten Sie, daß dieser Prozessor Instruktionen aufweist, die Operanden in bestimmten Registern erwarten. Dies ist mittels "vorgefärbter" Intervalle zu berücksichtigen, denen bereits zu Beginn der Allokation ein Register zugewiesen ist.
- Die Floating-Point-Register werden auf dem Intel x86 kellerartig verwaltet. Passen Sie Ihr Verfahren bzw. die anschließende Codeerzeugung so an, daßbei jedem Zugriff auf ein

FPU-Register dieses am Kellerende steht und daher in der Instruktion benutzt werden kann.

- Erweitern Sie den Codegenerator des Compilers so, daß die in der SSE2-Architektur des Intel-Prozessors definierten Instruktionen erzeugt werden können, die keinen kellerartigen Zugriff auf die FPU-Register erfordern.

Dokumentieren Sie Ihre Algorithmen und Datenstrukturen sorgfältig und auf abstraktem Niveau. Führen Sie Zeitmessungen durch, die den ursprünglichen Compiler und Ihren neuen Compiler hinsichtlich Laufzeit und Effizienz des erzeugten Codes vergleichen. Fassen Sie wenn möglich Ihre Diplomschrift auf Englisch ab, damit sie von Compilerbauern bei Sun Microsystems verstanden werden kann.

Der Fortgang der Arbeit soll in 14-tägigem Abstand mit dem Betreuer besprochen werden. Für die Ausarbeitung der schriftlichen Diplomarbeit sind die Richtlinien der Abteilung Systemsoftware zu beachten.

Betreuer: o.Univ.-Prof. Dr. Hanspeter Mössenböck

Ausgabe: 1. Oktober 2003