Fakultät für Informatik und Mathematik
Dr. Christoph Herrmann

WS 2006/2007

Übungen zur Vorlesung: Struktur und Implementierung von Programmiersprachen II

Blatt 8 (Codegenerierung für Basisblöcke)

Aufgabe 13 (DAG-Generierung, Leitersequenzen und Register-Tracking)

Gegeben ist ein Basisblock aus einem C-Programm:

```
{
   int x, z;
   z = y*b;
   y = c+z;
   x = z+3;
   z = y-x;
   c = y*z;
   y = x*d;
}
```

Die in einigen Teilaufgaben vorkommenden Abbildungsverweise beziehen sich auf das Buch: "Modern Compiler Design". Bei den Teilaufgaben (f), (g) und (h) soll die Kommutativität von Addition und Multiplikation ausgenutzt werden.

(a) Geben Sie an, welche Variablen Eingangs- und Ausgangsvariablen sind.

(b) Zeichnen Sie, analog zu Abb. 4.42, den abstrakten Syntaxbaum für den Basisblock in schwarzer Farbe.

(c) Ergänzen Sie mit grüner Farbe die True-Abhängigkeiten und mit blauer Farbe die Anti- und Output-Abhängigkeiten. Kreuzen Sie mit roter Farbe die entfallenden Kontrollabhängigkeitskanten des abstrakten Syntaxbaums durch.

(d) Verwenden Sie die Tripeldarstellung wie in Abb. 4.48, um einen gerichteten azyklischen Graphen (DAG) ohne Mehrfachvorkommen gemeinsamer Teilausdrücke zu erzeugen.

(e) Zeichnen Sie den DAG, wobei die inneren Knoten neben dem Operator auch mit der Position des sie berechnenden Tripels nummeriert sind.

(f) Kennzeichnen Sie in der DAG-Graphik die maximalen Leitersequenzen mittels umhüllender Kurven und nummerieren Sie diese. Annotieren Sie Knoten, deren Wert in einem Pseudoregister (Temporary) gespeichert werden muss, mit einem Namen für dieses Pseudoregister.

(g) Geben Sie den Zielcode unter Verwendung der Pseudoregister an. Dabei soll, analog zu dem Code in Abb. 4.53, folgende Erweiterung des Maschinenbefehlssatzes aus Abb. 4.22 verwendet werden: der zweite Operand bei den Operationen `Add`, `Subtr` und `Mult` kann auch eine Konstante (Suffix: `Const`) oder der Inhalt einer Speicherzelle (Suffix: `Mem`) sein. Ausserdem kann mit dem Befehl `Load_Reg` der Inhalt eines Registers in den eines anderen Registers kopiert werden. Markieren Sie für jede Leitersequenz den zu ihr korrespondierenden Codeabschnitt unter Angabe ihrer Nummer.

(h) Nehmen Sie an, dass alle Pseudoregister realen Registern zugewiesen werden und optimieren Sie dann den Code mittels Register-Tracking. Es soll nur soviele Speicheroperationen geben wie für das Lesen der Eingangsvariablen und das Schreiben der Ausgangsvariablen unbedingt notwendig sind, wenn die Umgebung des Basisblocks wie hier unbekannt ist; dabei soll angenommen werden, dass die Inhalte der verwendeten Register von der Umgebung des Basisblocks nicht mehr gebraucht werden.

Das vorrangige Ziel soll die Minimierung der Anzahl verwendeter Register sein, das zweite (nur bei gleicher Registerzahl) die Minimierung der Codelänge.

Besprechung: in der Übung am Donnerstag, 21.12.2006.