P. Thiemann, M. Neubauer Wintersemester 2004/2005

Compilerbau

http://proglang.informatik.uni-freiburg.de/teaching/compilerbau/2004/

Übungsblatt 12

Abgabe: 9.02.2005

Aufgabe 1 (Codegenerierung von Tail-Calls):

Der in der Vorlesung vorgestellte generierte Code für Tail-Calls ist insofern suboptimal, dass bei Tail-Calls zuerst der aktuellste *activation record* entfernt und anschliessend durch einen neuen für die aufgerufene Funktion ersetzt wird, bevor endgültig zum Funktioncode gesprungen wird.

Optimiere die Codeerzeugung für Tail-Calls, indem der aktuellste *activation record* wiederverwendet wird. Füge dazu für Funktionen eventuell einen zweiten Eintrittspunkt ein, der ausschließlich für Tail-Calls verwendet wird.

Aufgabe 2 (Liveness-Analyse):

Führe eine Flussanalyse für das folgende Pseudo-Assemblerprogramm durch:

```
 1  m ← 1
 2  v ← 0
 3  if v ≥ n goto 15
 4  r ← v
 5  s ← 0
 6  if r < n goto 9
 7  v ← v + 1
 8  goto 3
 9  x ← MEM[r]
10  s ← s + x
11  if s ≤ m goto 13
12  m ← s
13  r ← r + 1
14  goto 6
15  return m
```

Gehe dazu wie folgt vor:

(i) Zeichne den Kontrollflussgraphen (CFG) des Programs

(ii) Berechne die *live-in*- und *live-out*-Mengen jeder Anweisung

(iii) Erstelle den *register interference graph*

Aufgabe 3 (Liveness-Analyse):

Zeige, dass

(i) die Datenfluss-Gleichungen zur Liveness-Berechnung aus der Vorlesung einen kleinsten Fixpunkt besitzen, und dass

(ii) der zugehörige Algorithmus aus der Vorlesung gerade diesen kleinsten Fixpunkt berechnet.