P. Thiemann, M. Neubauer Wintersemester 2004/2005

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

Übungsblatt 11

Abgabe: 2.02.2005

Aufgabe 1 (optimale Codegenerierung):

`IBurg`[1] ist ein Codegeneratorgenerator für *trivial term templates*, der Bottom-up-Codegeneratoren in C erzeugt.

Die folgende IBurg-Spezifikation gibt Baum-Templates mit Kosten für Kombinationen aus vier Instruktionen und Konstanten an:

```
%term MOVE=1 MEM=2 PLUS=3 NAME=4 CONST=6
%%
stm:    MOVE(MEM(loc),reg) = 1 (4);

reg:    PLUS(con,reg) = 2 (3);
reg:    PLUS(reg,reg) = 3 (2);
reg:    PLUS(MEM(loc),reg) = 4 (4);
reg:    MEM(loc) = 5 (4);
reg:    con = 6 (2);

loc:    reg = 7;
loc:    NAME = 8;
loc:    PLUS(NAME,reg) = 9;

con:    CONST = 10;
%%
```

Die erste Zeile spezifiert die Namen aller verwendeten Baumknoten-Labels. Die dann folgenden Regeln beginnen jeweils mit einem Nonterminal und enthalten weiter ein Baumpattern, ein Gleichheitszeichen (=), die Nummer einer assoziierten Aktion (die hier ignoriert werden kann) und optional die assoziierten Kosten (in Klammern). Falls keine Kosten angegeben sind, sind diese als 0 anzusehen.

Rechne eine global-optimale Überdeckung für den folgenden Ausdrucksbaum aus:

```
MOVE(MEM(NAME),PLUS(MEM(PLUS(NAME,MEM(NAME),CONST))))
```

Aufgabe 2 (Erstellen einer IBurg-Spezifikation):

Entwerfe einen einfachen Codegenerator, der MIPS-Code[2] für triviale Ausdrucksbäume mit Interger-Arithmetik (inkl. Integer-Konstanten, Speicherzugriff, Registerzugriff, Addition, Subtraktion, Multiplikation und Division) erzeugt.

[1] `http://www.cs.princeton.edu/software/iburg/`

[2] Eine Übersicht über den MIPS-Instruktionssatz findet man etwa unter `http://www.mrc.uidaho.edu/mrc/people/jff/digital/MIPSir.html`.

Erstelle dazu eine IBurg-Spezifikation (mit passenden Kosten), um eine optimale Überdeckungen zu ermittelen und gibt passende semantische Aktionen an, um die ausgewählten MIPS-Instruktionen auszugeben.