Software-Modellierung und Verifikation
Informatik 2
Prof. J.-P. Katoen
RWTH Aachen

Priv.-Doz. T. Noll noll@cs.rwth-aachen.de
D. Klink klink@cs.rwth-aachen.de

10. Exercise sheet *Compiler Construction 2008*

Due to Wed., 16 July 2008. Bring it to the chair.

The exercises on this sheet are taken from an earlier exam. You might try to first prepare yourself and then solve the exercises without having a look into your learning material. All exercises are optional such that those who feel they are prepared for the exam do not have to solve them to get the points. However, if you still need some points, this is your opportunity!

Exercise 10.1: (optional)

a) Give a regular expression, describing the language of all rational numbers with at least one decimal place, but without superfluous zeros. I.e. 0.0 and 1.001 are in the language, but not $\underline{0}0.0$, $1.0\underline{00}$ and 1 (because there is no decimal place at all).

b) Construct an NFA that recognises the language from a). Superfluous ε-transitions may be omitted.

c) Give the chain of sets of states that are visited when recognizing 0.01 using the NFA method.

Exercise 10.2: (optional)

Consider the following grammar:

$$\begin{array}{lcl} S & \rightarrow & S'S \mid S' \\ S' & \rightarrow & a \mid (T') \\ T' & \rightarrow & S'T' \mid S' \end{array}$$

a) Transform the grammar into an equivalent $LL(1)$-grammar G'.

b) Show that $G' \in LL(1)$.

Exercise 10.3: (optional)

Consider the following grammar:

$$\begin{array}{lcl} S & \rightarrow & A \\ A & \rightarrow & BdC \\ B & \rightarrow & Ba \mid Bb \mid a \\ C & \rightarrow & Ca \mid Cb \mid a \end{array}$$

a) Construct the $LR(0)$ sets for the grammar.

b) Show that the grammar is in $SLR(1)$ but not in $LR(0)$.

Exercise 10.4: (optional)

Consider the following attribute grammar:

$$
\begin{array}{llll}
S \rightarrow A & \beta.1 = \alpha.1 & A \rightarrow a & \alpha.0 = \beta.0 \\
A \rightarrow AB & \alpha.0 = \alpha.1 & B \rightarrow b & \alpha.0 = \beta.0 \\
 & \beta.2 = \beta.0 & & \\
 & \beta.1 = \alpha.2 & &
\end{array}
$$

a) Check whether this is an S-attributed grammar.

b) Check whether this is an L-attributed grammar.

Exercise 10.5: (optional)

Translate the following EPL-program into intermediate code.

in/out x;

while $(x < 0)$ **do**

$x := x + 1$;

Exercise 10.6: (optional)

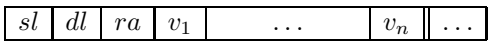
Consider the following intermediate code:

```
      ⋮
 7:   LOAD(1, 2);       (dif, off)
 8:   ADD;
 9:   RET;
      ⋮
26:   CALL(38, 1, 3);   (ca, dif, loc)
```

Give the next four states of the abstract machine starting in:

$$(ca, d, p) := (7, -3, 9 : 4 : 26 : 3 : 7 : 4 : 3 : 36 : 5 : 10 : 4 : 40 : 1 : 2 : \ldots)$$

Recall that the procedure stack has the form:



and the *base*-function is defined as:

$$
\begin{array}{rcl}
base(p, 0) & := & 1 \\
base(p, dif + 1) & := & base(p, dif) + p.base(p, dif)
\end{array}
$$