Universität Karlsruhe (TH)

Institut für Innovatives Rechnen und Programmstrukturen (IPD)

Übersetzerbau WS 2003/04 http://www.info.uni-karlsruhe.de/

Dozent: Prof. Dr.rer.nat. G. Goos goos@ipd.info.uni-karlsruhe.de

Übungsleiter: Rubino Geiß rubino@ipd.info.uni-karlsruhe.de

Übungsblatt 3-Zusatz Ausgabe: 17.11.2003 Besprechung: 20.11.2003

Alle Aufgaben sollten mit Hilfe von gängigen LALR-Zerteilergeneratoren überprüfbar sein. Wir haben uns für das "lalr"-Werkzeug aus "cocktail" entschieden. Das Werkzeug steht bereit auf:

`ftp://i44ftp.info.uni-karlsruhe.de/pub/cocktail`

Aufgabe 1: *Rechtsfaktorisieren, Shift-Reduktionskonflikt*

Geben Sie für die folgende Grammatik eine sprachgleiche Grammatik an, die keine Verletzungen der $LALR(1)$-Bedingung enthält. Bei Ihrer Konstruktion beachten Sie bitte die abgedruckten Ein- und Ausgaben des "lalr"-Werkzeugs.

$$S \to Xbb, S \to Xbbc, X \to Xb, X \to b$$

Die obige Grammatik sieht als Eingabespezifikation des "lalr"-Werkzeugs folgendermaßen aus:

```
TOKEN   'b'  'c'

RULE
S       : X 'b' 'b'
        | X 'b' 'b' 'c' .

X       : X 'b'
        | 'b' .
```

Beim Aufruf (lalr -c -b -v aufg1.lalr) werden verschiedene Fehlermeldungen generiert (auf stderr und _Debug):

```
Information state is not LALR(1) - state 4
Warning     automatically repaired read reduce conflict on  'b'
  1 warning(s)  1 information(s)


State 4


S _EndOfFile
X 'b' 'b'
:
X 'b'
:...........
           :
reduce X -> X 'b'. {'b'} ?
```

```
S _EndOfFile
X 'b' 'b'
:............
           :
read   S -> X 'b'.'b' ?

S _EndOfFile
X 'b' 'b' 'c'
:............
           :
read   S -> X 'b'.'b' 'c' ?


retained               S -> X 'b'.'b'
ignored                X -> X 'b'. {'b'}
retained               S -> X 'b'.'b' 'c'
```

Aufgabe 2: *Rechtsfaktorisieren, Reduktions-Reduktionskonflikt*

Konstruieren Sie zu der folgenden Grammatik eine sprachgleiche Grammatik, die $LALR(1)$ ist. Bei Ihrer Konstruktion beachten Sie bitte die Fehlermeldungen des "lalr"-Werkzeugs.

```
TOKEN  'b'  'c'

RULE S
        : X  |  Y X  .

X       : X 'b' |  'b'  .

Y       : Y 'b'  |  'c'  .
```

Beim Aufruf (lalr -c -b -v aufg2.lalr) werden verschiedene Fehlermeldungen generiert (auf stderr und _Debug):

```
Information state is not LALR(1) - state 9 Warning
automatically repaired reduce reduce conflict on  'b'
  1 warning(s)  1 information(s)


State 9


S _EndOfFile X X 'b' : 'b' :............
           :
reduce X -> 'b'. {'b'} ?


S _EndOfFile Y X : 'b' : Y 'b' :............
           :
reduce Y -> Y 'b'. {'b'} ?


retained               X -> 'b'. {'b'}
ignored                Y -> Y 'b'. {'b'}
```

Aufgabe 3: *Precedenz*

Wie könnte man folgendes Grammatikfragment LALR(1) machen?

```
RULE statement
                : 'IF' expression 'THEN' statement
                | 'IF' expression 'THEN' statement 'ELSE' statement
                | '...'  .

expression      : 'TRUE'
                | 'FALSE'  .
```

Die Fehlermeldung ist:

```
Information state is not LALR(1) - state 8 Warning
automatically repaired read reduce conflict on  'ELSE'
  1 warning(s)  1 information(s)


State 8


statement _EndOfFile 'IF' expression 'THEN' statement 'ELSE'
statement
                        :
                        'IF' expression 'THEN' statement
   reduce statement -> 'IF' expression 'THEN' statement. {'ELSE'} ?
   read   statement -> 'IF' expression 'THEN' statement.'ELSE' statement ?


ignored        statement -> 'IF' expression 'THEN' statement. {'ELSE'}
retained       statement -> 'IF' expression 'THEN'statement.'ELSE' statement
```