FernUniversität in Hagen

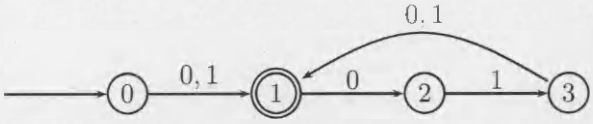
Lösungsvorschläge zur Hauptklausur „1810 Übersetzerbau“

03.03.2018

Aufgabe 1

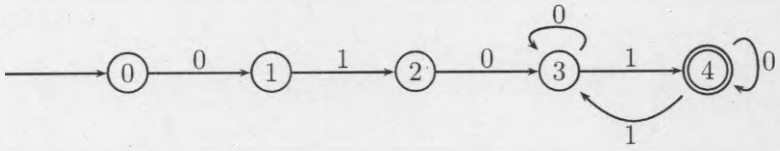
(a)

Der folgende Automat akzeptiert L_1:



(b)

Der folgende Automat akzeptiert L_2:

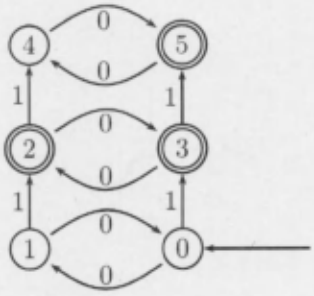


(c)

$r_3 = (1|01|001|0001)^*(0|00|000|\varepsilon)$.

(d)

Der folgende Automat akzeptiert L_4:



Aufgabe 2

(a)

Wir beseitigen zunächst die Linksrekursion in der Produktion $B \to B\mathbf{yzy}A \mid A\mathbf{zy}$. Als Resultat erhalten wir die Grammatik $G' = (\{S, A, B, C\}, \{\mathbf{w}, \mathbf{x}, \mathbf{y}, \mathbf{z}\}, P, S)$ mit den Produktionen

$$\begin{aligned} P = \{S &\to A, \\ A &\to \mathbf{wx}B\mathbf{w} \mid \mathbf{x}, \\ B &\to A\mathbf{zy}C, \\ C &\to \mathbf{yzy}AC \mid \varepsilon\}. \end{aligned}$$

(b)

Durch Berechnung der FIRST- und FOLLOW-Mengen erhalten wir die folgenden Steuermengen:

Nr.	Produktion	Steuermenge
(1)	$S \to A$	$\{\mathbf{w}, \mathbf{x}\}$
(2)	$A \to \mathbf{wx}B\mathbf{w}$	$\{\mathbf{w}\}$
(3)	$A \to \mathbf{x}$	$\{\mathbf{x}\}$
(4)	$B \to A\mathbf{zy}C$	$\{\mathbf{w}, \mathbf{x}\}$
(5)	$C \to \mathbf{yzy}AC$	$\{\mathbf{y}\}$
(6)	$C \to \varepsilon$	$\{\mathbf{w}\}$

(c)

Unter Verwendung des allgemeinen Schemas aus dem Kurstext erhalten wir die folgenden Prozeduren:

```
prodecure S;
begin
  if symbol = w or symbol = x then
    output(1); bearbeite A;
  else error;
  fi
end;

prodecure A;
begin
  if symbol = w then
    output(2); match(w); match(x); bearbeite B; match(w);
  elsif symbol = x then
    output(3); match(x);
  else error;
  fi
end;
```

```
prodecure B;
begin
  if symbol = w or symbol = x then
    output(4); bearbeite A; match(z); match(y); bearbeite C;
  else error;
  fi
end;

prodecure C;
begin
  if symbol = y then
    output(5); match(y); match(z); match(y); bearbeite A; bearbeite C;
  elsif symbol = w then
    output(6);
  else error;
  fi
end;
```

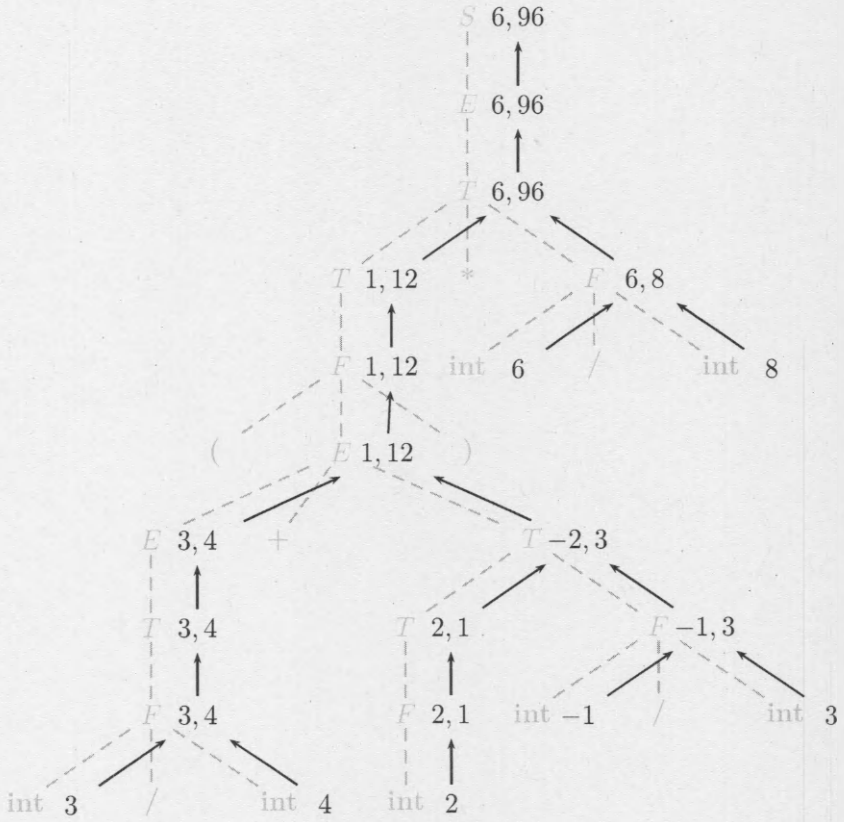
Aufgabe 3 Attributierte Grammatik

(a)

Wir führen zwei Attribute *nom* und *denom* ein, die den Zähler bzw. den Nenner des Bruchs darstellen. Wir erhalten folgende semantische Definition:

Produktion	*Semantische Regel*
$S \rightarrow E$	$S.nom := E.nom$ $S.denom := E.denom$
$E \rightarrow E_1 + T$	$E.nom := E_1.nom * T.denom + T.nom * E_1.denom$ $E.denom := E_1.denom * T.denom$
$E \rightarrow T$	$E.nom := T.nom$ $E.denom := T.denom$
$T \rightarrow T_1 * F$	$T.nom := T_1.nom * F.nom$ $T.denom := T_1.denom * F.denom$
$T \rightarrow F$	$T.nom := F.nom$ $T.denom := F.denom$
$F \rightarrow (E)$	$F.nom := E.nom$ $F.denom := E.denom$
$F \rightarrow \mathbf{int}$	$F.nom := \mathbf{int}.v$ $F.denom := 1$
$F \rightarrow \mathbf{int}_1 / \mathbf{int}_2$	$F.nom := \mathbf{int}_1.v$ $F.denom := \mathbf{int}_2.v$

(b)



(c)

Linksrekursion gibt es bei den Produktionen für die Symbole E und T. Wir erhalten folgende semantische Definitionen für diese Produktionen:

Produktion	*Semantische Regel*
$E \rightarrow T$	$R.v.nom := T.nom$
	$R.v.denom := T.denom$
R	$E.nom := R.s.nom$
	$E.denom := R.s.denom$
$R \rightarrow + T$	$R_1.v.nom := R.v.nom * T.denom + T.nom * R.v.denom$
	$R_1.v.denom := R.v.denom * T.denom$
R_1	$R.s.nom := R_1.s.nom$
	$R.s.denom := R_1.s.denom$
$R \rightarrow \epsilon$	$R.s.nom := R.v.nom$
	$R.s.denom := R.v.denom$
$T \rightarrow F$	$U.v.nom := F.nom$
	$U.v.denom := F.denom$
U	$T.nom := U.s.nom$
	$T.denom := U.s.denom$
$U \rightarrow * F$	$U_1.v.nom := U.v.nom * F.nom$
	$U_1.v.denom := U.v.denom * F.denom$
U_1	$U.s.nom := U_1.s.nom$
	$U.s.denom := U_1.s.denom$
$U \rightarrow \epsilon$	$U.s.nom := U.v.nom$
	$U.s.denom := U.v.denom$

Aufgabe 4

(a)

Der Term $\cos(t)$ kann aus beiden Schleifen ausgelagert werden, der Term $i - s$ aus der inneren. Wir führen hierzu die Hilfsvariablen u und v ein.

```
i := 0;
u := cos(t);
w := m + 1;
while i < n do
  v := i - s;
  for j := i to w
    x := j * u;
    y := v;
    output(x, y);
  end
  i := i + 2;
end
```

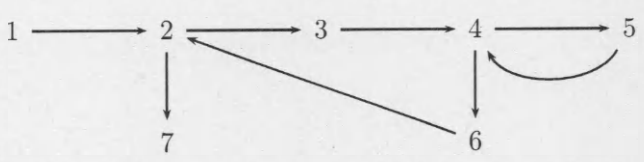
(b)

Wir geben den 3-Adress-Code zu dem Programmstück direkt mit seiner Unterteilung in Basisblöcke an.

Block		Anweisung
1	(1)	`i := 0`
	(2)	`u := cos(t)`
	(3)	`t := m + 1`
2	(4)	`if i >= n then goto (15)`
3	(5)	`v := i - s`
	(6)	`j := i`
4	(7)	`if j > t then goto (13)`
5	(8)	`x := j * u`
	(9)	`y := v`
	(10)	`output(x, y)`
	(11)	`j := j + 1`
	(12)	`goto (7)`
6	(13)	`i := i + 2`
	(14)	`goto (3)`
7	(15)	`...`

(c)

Der Flussgraph zu dem 3-Adress-Code aus Teil (b) hat die folgende Gestalt:



(d)

Die Multiplikation in Zeile (8) lässt sich in Additionen überführen. Dazu führen wir die Hilfsvariable w ein.

Block		Anweisung
1	(1)	`i := 0`
	(2)	`u := cos(t)`
	(3)	`t := m + 1`
2	(4)	`if i >= n then goto (17)`
3	(5)	`v := i - s`
	(6)	`j := 1`
	(7)	`w := j * u`
4	(8)	`if j > t then goto (15)`
5	(9)	`x := w`
	(10)	`y := v`
	(11)	`output(x, y)`
	(12)	`j := j + 1`
	(13)	`w := w + u`
	(14)	`goto (7)`
6	(15)	`i := i + 2`
	(16)	`goto (3)`
7	(17)	...