

Syntaxanalyse: LR-Parser (Teil 2)

BC George (FH Bielefeld)

Unless otherwise noted, this work is licensed under CC BY-SA 4.0.

Wiederholung

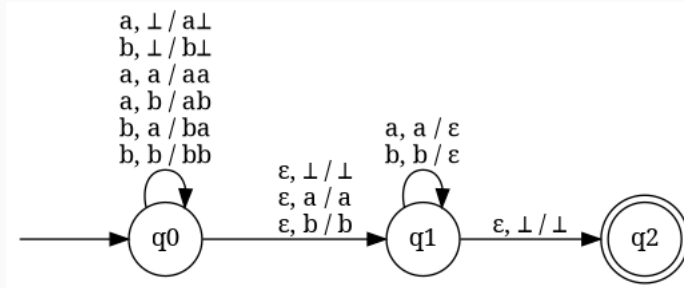
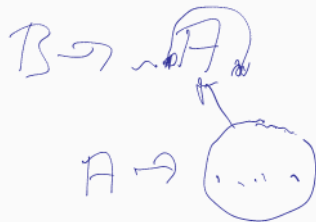
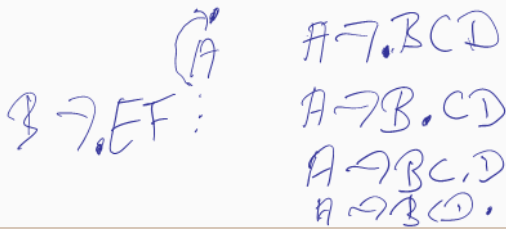


Abbildung 1: Ein PDA für $L = \{ww^R \mid w \in \{a, b\}^*\}$

Top-Down-Analyse



- LL reicht nicht.
- LR: Aufbau des Ableitungsbaums von unten nach oben.
- LR(0): Parsen ohne Vorschautoken
- Ein DFA mit einem Stack wird über eine Tabelle mit Aktions- und Sprungbefehlen gesteuert.
- Im Stack stehen Zustände des DFAs.
- Diese Zustände werden mit sog. Dotted Items und deren Closures identifiziert.



$B \rightarrow EF$

$A \rightarrow BCD$

$A \rightarrow B.CD$

$A \rightarrow BC.D$

$A \rightarrow BCD.$

Motivation

Auch $LR(0)$ ist nicht alles

Die Menge der $LR(0)$ -Sprachen ist eine echte Teilmenge der deterministisch kontextfreien Sprachen. Wir brauchen ein Verfahren, mit dem man alle deterministisch kontextfreien Sprachen parsen kann.

Bottom-Up-Analyse mit Vorschautoken

LR-Parsen mit 1 Vorschautoken

Ist eine Grammatik nicht LR(0), kann sie vielleicht mit einem Vorschautoken geparkt werden. Hier gibt es drei Verfahren:

- SLR(1)-Parsing
- (kanonisches) LR(1)-Parsing
- LALR(1)-Parsing

in der Tabelle

| | <u>a</u> | <u>b</u> | <u>c</u> | <u>d</u> | <u>e</u> |
|-------|----------|----------|----------|----------|----------|
| q_1 | r_2 | r_2 | r_2 | r_2 | r_2 |
| | | r_5 | | r_6 | r_7 |

$A \rightarrow BCD$

$E \rightarrow BCDa$

$A \begin{cases} B \\ C \\ D \end{cases}$

$G \begin{cases} B \\ C \\ D \end{cases}$



SLR

Simple LR(1) = (SLR)-Parsing



$A \rightarrow \beta$ wird nur reduziert, wenn das Vorschautoken in der *FOLLOW*-Menge von A ist.

⇒ Es ändert sich nur die Parse Table:

Bei allen LR(0)-Items in der Tabelle, die einen Punkt am Ende der rechten Seite stehen haben, trage in der Aktionstabelle beim zugehörigen Zustand die Reduktion mittels der zugehörigen Regel bei allen Terminals ein, die in der FOLLOW-Menge des Nonterminals auf der linken Seite der Regel enthalten sind.

Der SLR-Automat der Grammatik G1:

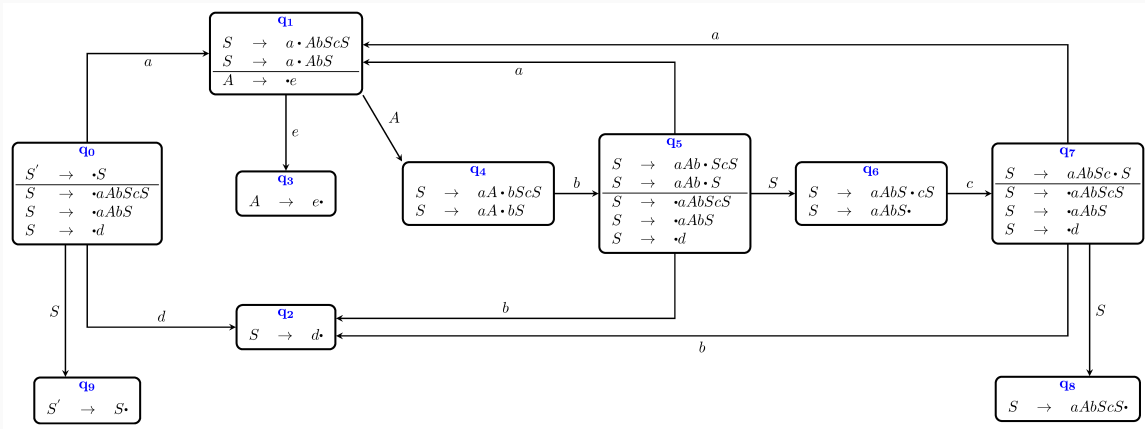


Abbildung 2: SLR(1)-Automat

Die SLR-Parsertabelle der Grammatik G1

Vorschauaktionen

| Zustand | Aktion | | | | | | Sprung | |
|---------|----------|----------|----------|----------|----------|---------|----------|----------|
| | <i>a</i> | <i>b</i> | <i>c</i> | <i>d</i> | <i>e</i> | \perp | <i>S</i> | <i>A</i> |
| q_0 | s1 | | | s2 | | | q_9 | |
| q_1 | | | | | s3 | | | q_4 |
| q_2 | | | r3 | | | r3 | | |
| q_3 | | r4 | | | | | | |
| q_4 | | s5 | | | | | | |
| q_5 | s1 | s2 | | | | | q_6 | |
| q_6 | | | s7,r2 | | | r2 | | |
| q_7 | s1 | s2 | | | | | q_8 | |
| q_8 | | | r1 | | | r1 | | |
| q_9 | | | | | | acc | | |

Abbildung 3: SLR(1)-Parsertabelle

Zum Vergleich: Die LR(0)-Tabelle von G1 (letzte Vorlesung)





| Zustand | Aktion | | | | | | Sprung | |
|---|-----------|----------|--------------|----------|----------|---------|-----------------------|-----------------------|
| | <i>a</i> | <i>b</i> | <i>c</i> | <i>d</i> | <i>e</i> | \perp | <i>S</i> | <i>A</i> |
| <i>q</i> ₀ | s1 | | | s2 | | | <i>q</i> ₉ | |
| <i>q</i> ₁ | | | | | s3 | | | <i>q</i> ₄ |
|  <i>q</i> ₂ | r3 | r3 | r3 | r3 | r3 | r3 | | |
|  <i>q</i> ₃ | r4 | r4 | r4 | r4 | r4 | r4 | | |
| <i>q</i> ₄ | | s5 | | | | | | |
| <i>q</i> ₅ | s1 | s2 | | s2 | | | <i>q</i> ₆ | |
|  <i>q</i> ₆ | <u>r2</u> | r2 | <u>s7,r2</u> | r2 | r2 | r2 | | |
| <i>q</i> ₇ | s1 | s2 | | | | | <i>q</i> ₈ | |
|  <i>q</i> ₈ | r1 | r1 | r1 | r1 | r1 | r1 | | |
| <i>q</i> ₉ | | | | | | acc | | |

Abbildung 4: LR(0)-Parsertabelle

Kanonische LR(1)-Syntaxanalyse

Mehr geht nicht: Kanonische LR(1)-Syntaxanalyse = LR-Analyse

$$A \rightarrow \beta \quad , \quad \text{Follow}(A)$$

Beim SLR-Verfahren wird nach $A \rightarrow \beta$ reduziert, wenn das Vorschautoken in $\text{Follow}(A)$ liegt. Dabei kann es vorkommen, dass das Vorschautoken ein Element davon ist, aber genau bei dieser Regel kann es nicht dem A folgen. Es wird also falsch reduziert, und es entstehen zu viele Einträge in der Tabelle (Konflikte!).

Jetzt werden nicht Follow-Mengen von Nichtterminalen, sondern LOOKAHEAD-Mengen von Produktionen berechnet.

$S \Rightarrow A \alpha B$ $\text{Follow}(B) = \{a_1, \dots\}$

$A \rightarrow b B | c$

$B \rightarrow A$

$S \Rightarrow A \alpha B \Rightarrow b B \alpha B$

$B \rightarrow A$ hätte nicht ausgeführt werden dürfen


Vorschautoken

$\text{Follow}(A)$ nicht als Kriterium

Die LR(1)-Items

Zu jedem LR(0)-Item (hier auch *Kern* genannt) wird eine *LOOKAHEAD* - Menge L hinzugefügt, die angibt, welche Terminals dem Symbol auf der linken Seite folgen können.

z. B. $[S' \rightarrow \cdot S, \{\perp\}]$



Die Hülle $CLOSURE_1$

1. füge I zu $CLOSURE_1(I)$ hinzu
2. gibt es ein LR(1) - Item $[A \rightarrow \alpha \cdot B\beta, L]$ aus $CLOSURE_1(I)$ und eine Produktion $(B \rightarrow \gamma)$, füge $[B \rightarrow \cdot\gamma, FIRST(\beta L)]$ zu $CLOSURE_1(I)$ hinzu (α, β dürfen ϵ sein).

schon da

$GOTO_1(I, X)$ = eine Produktion

$CLOSURE_1(\{[A \rightarrow \alpha X \cdot \beta, L] \mid [A \rightarrow \alpha \cdot X \beta, L] \in I\})$

für eine Itemmenge I und $X \in N \cup T, A \in N, \alpha, \beta \in (N \cup T)^*$.

Der LR(1)-Automat

Der Automat wird analog zum LR(0)-Automaten erstellt mit dem Startzustand

$$[S' \rightarrow \cdot S, \{\perp\}]$$

Die Tabelle unterscheidet sich nur bei der Reduktion von der LR(0)-Tabelle:

Reduktionsoperationen werden in den Spalten der Terminals eingetragen, die in der LOOKAHEAD-Menge der entsprechenden Regel enthalten sind.

Die Beispielgrammatik G2

$$(0) S' \rightarrow S \text{ } \textcolor{blue}{\downarrow}$$

$$(1) S \rightarrow NN$$

$$(2) N \rightarrow 0N$$

$$(3) N \rightarrow 1$$

Der LR(1)-Automat der Grammatik G2

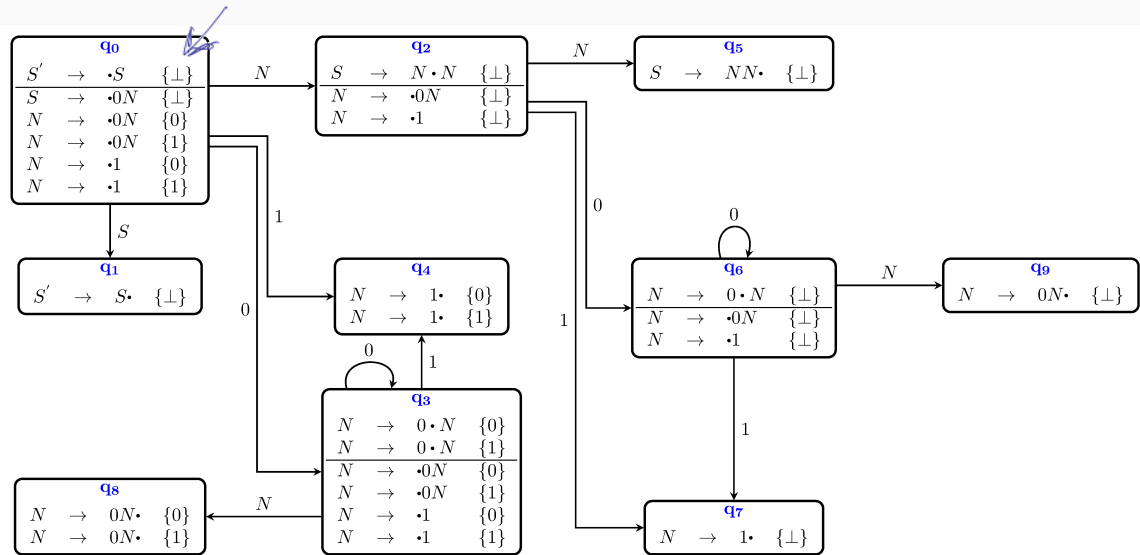



Abbildung 5: LR(1)-Automat

Die LR(1)-Parsertabelle der Grammatik G2



| Zustand | Aktion | | | Sprung | |
|---------|--------|----|---------|--------|-------|
| | 0 | 1 | \perp | S | N |
| q_0 | s3 | s4 | | q_1 | q_2 |
| q_1 | | | acc | | |
| q_2 | s6 | s7 | | | q_5 |
| q_3 | s3 | s4 | | | q_8 |
| q_4 | r3 | r3 | | | |
| q_5 | | | r1 | | |
| q_6 | s6 | s7 | | | q_9 |
| q_7 | | | r3 | | |
| q_8 | r2 | r2 | | | |
| q_9 | | | r2 | | |

Abbildung 6: LR(1)-Parsertabelle

Lookahead-LR = LALR

LALR(1)

Zusammenfassung aller LR(1)-Zustände, die sich nur in den LOOKAHEAD-Mengen unterscheiden
Parsergeneratoren generieren oft direkt aus einem LR(0)- einen LALR(1)-Zustands- Übergangsgraphen
durch Hinzufügen der LOOKAHEAD-Mengen.

Der LALR-Automat der Grammatik G2

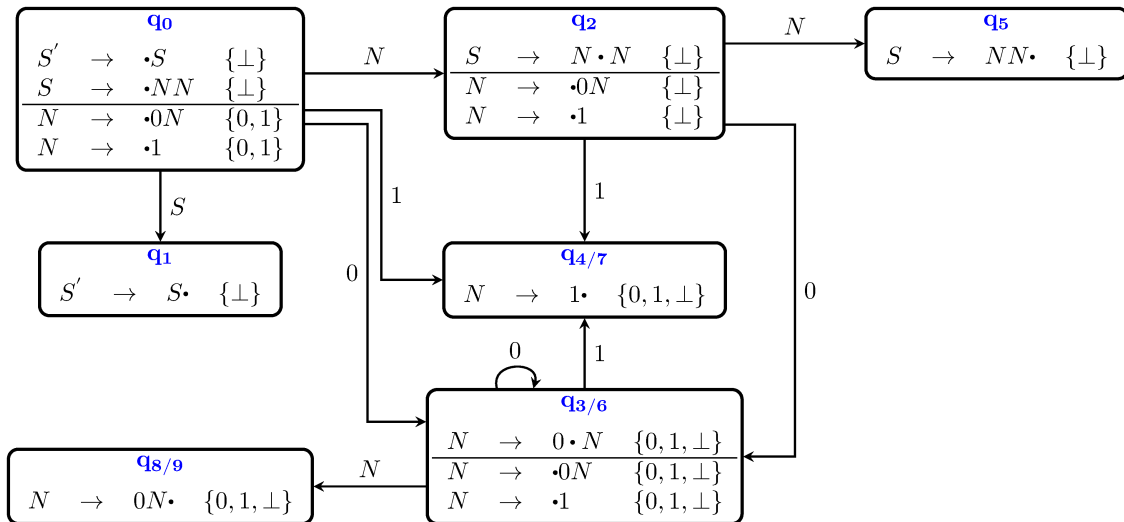


Abbildung 7: LALR(1)-Automat

Die LALR-Parsertabelle der Grammatik G2

| Zustand | Aktion | | | Sprung | |
|---------|--------|--------|---------|--------|---------|
| | 0 | 1 | \perp | S | N |
| q_0 | s(3/6) | s(4/7) | | q_1 | q_2 |
| q_1 | | | acc | | |
| q_2 | s(3/6) | s(4/7) | | | q_5 |
| $q_3/6$ | s(3/6) | s(4/7) | | | $q_8/9$ |
| $q_4/7$ | r3 | r3 | r3 | | |
| q_5 | | | r1 | | |
| $q_8/9$ | r2 | r2 | r2 | | |

Abbildung 8: LALR(1)-Parsertabelle

neuer Zustand

kein Konflikt

Bezug auf den neuen Zustand $q_8/9$

$k \geq 2$ **Vorschautoken**

Satz:

Zu jeder LR(k)-Sprache gibt es eine LR(1)-Grammatik.

Mehrdeutige Grammatiken

Es gibt auch Auswege

if B1
if B2
shiften ← (lke?? zu welchem if

Mehrdeutige Grammatiken sind oft leichter zu lesen und kleiner als die Grammatiken, die man erhält, wenn man die Mehrdeutigkeit auflöst, sofern möglich. Also die Grammatik mehrdeutig lassen!

Folgendes kann trotzdem helfen:

- Angabe von Vorrangregeln
- Angabe von Assoziativität
- Voreinstellung des Parsergenerators: z. B. Shiften bei Shift-Reduce-Konflikten
- Voreinstellung des Parsergenerators: z. B. Reduzieren nach der Regel, die in der Grammatik zuerst kommt bei Reduce-Reduce-Konflikten

$$a^b{}^c = a^{(b^c)} \text{ erst shiften}$$

Hierarchie der kontextfreien Sprachen

Hierarchie der kontextfreien Sprachen

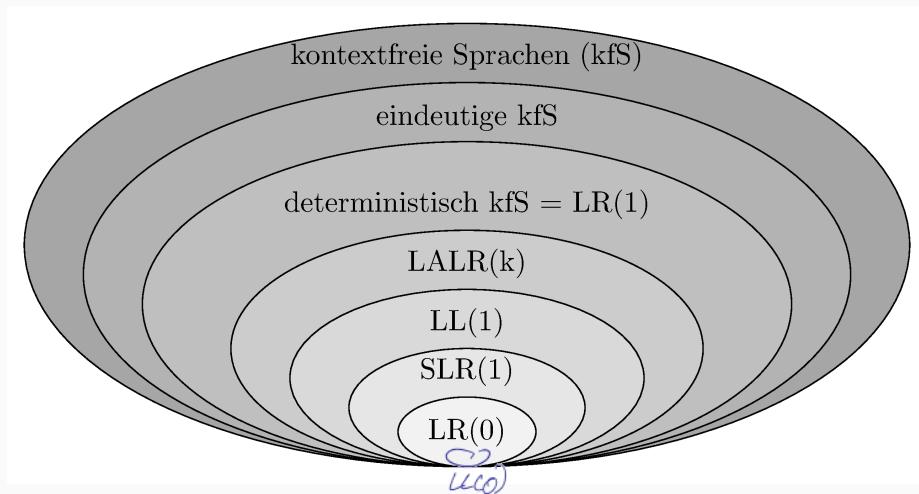


Abbildung 9: Sprachenhierarchie

Wrap-Up

mit DPDA

- mit Bottom-Up-Parsing LR(1) kann man alle deterministisch kontextfreien Sprachen parsen
- ein Vorschautoken genügt
- LR(0)-, SLR- und LALR- Parsing sind vereinfachte Verfahren für Teilmengen der LR-Sprachen

LICENSE



Unless otherwise noted, this work is licensed under CC BY-SA 4.0.