Übungsblatt 4 – Musterlösung

4.1

Der Mann holte die Kiste mit der Leiter aus dem Regal.

(a) Lesarten (u.a.):

1. Der Mann nutzte die Leiter, um die Kiste zu holen, die im Regal stand.
2. Der Mann nahm die Leiter aus dem Regal und holte dann mit dieser Leiter die Kiste.
3. Der Mann holte eine Kiste, in der sich eine Leiter befindet, aus dem Regal.
4. Der Mann holte die Kiste, in der sich die Leiter befand, die wiederum aus dem Regal stammte.

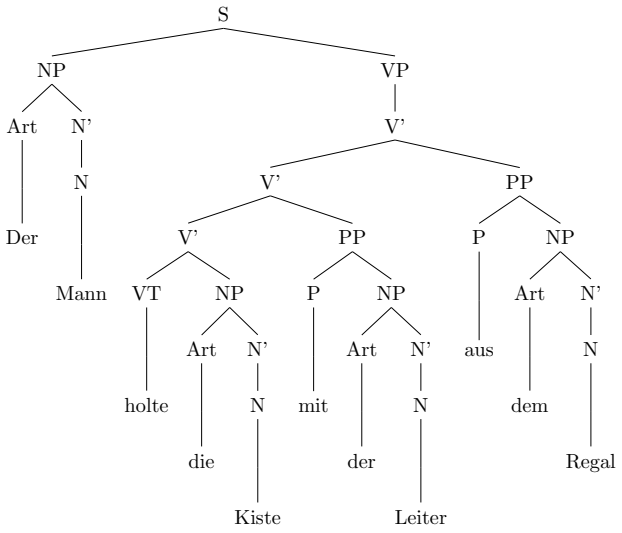
(b) Wahrscheinlichkeit der Lesarten

Einige Lesarten sind aus semantischer Sicht recht unwahrscheinlich, etwa dass es sich um eine riesige Kiste handelt, die eine Leiter enthalten kann. Oder dass die Leiter erst aus dem Regal genommen werden musste, um an einem unbestimmten Ort die Kiste zu erlangen.

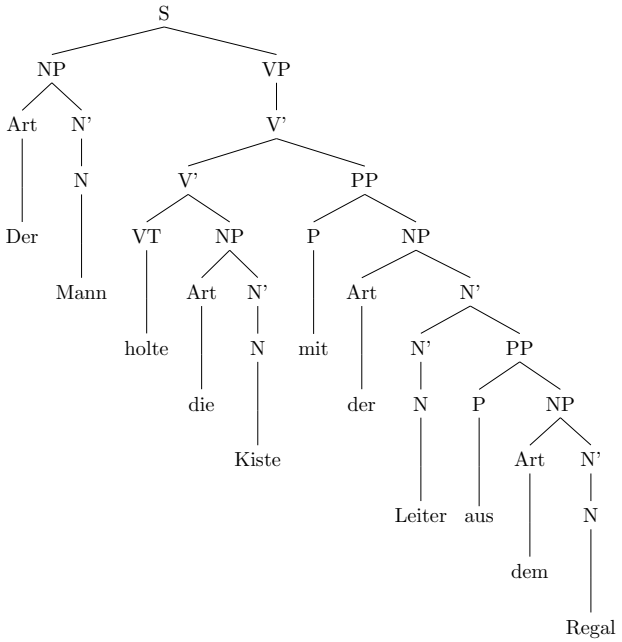
Um diese Lesarten als unwahrscheinlich zu beurteilen, bräuchte ein Computerprogramm sowohl wissen über die Semantik der einzelnen Wörter als auch Weltwissen. Beides ist einem einfachen Parser allerdings nicht zugänglich.

(c) Syntaxbäume

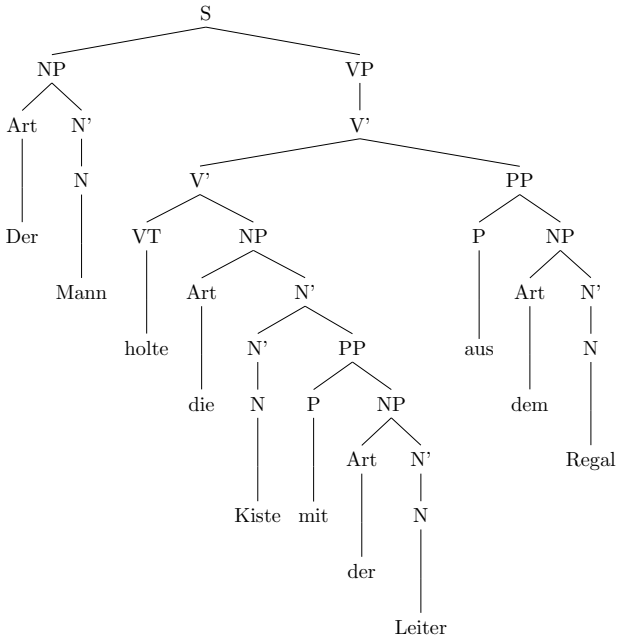
1.



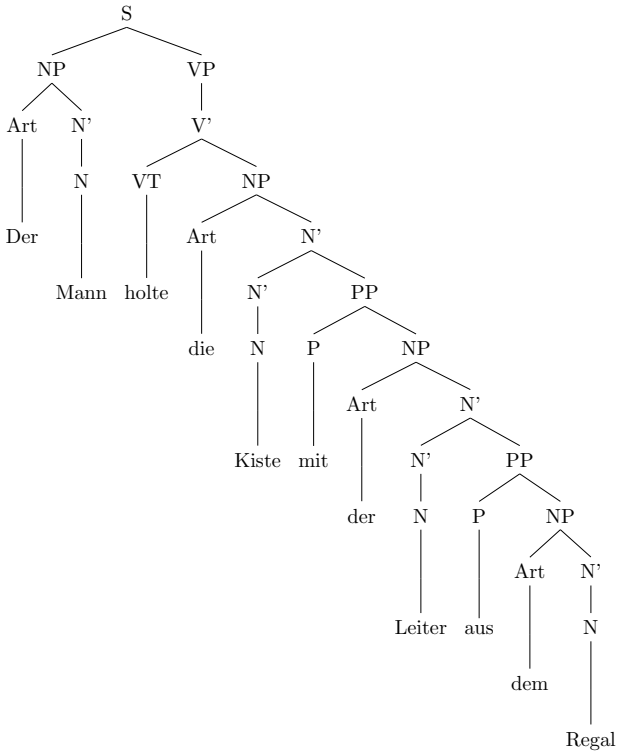
2.



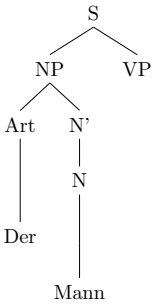
3.


S
NP
VP
Art
N'
V'
N
V'
PP
Der
VT
NP
P
NP
Mann
Art
N'
Art
N'
holte
N'
PP
aus
N
die
N
P
NP
dem
Art
N'
Regal
Kiste
mit
N
der
Leiter

4.


S
NP
VP
Art
N'
V'
N
VT
NP
Der
Art
N'
Mann
holte
N'
PP
die
N
P
NP
Art
N'
Kiste
mit
N'
PP
der
N
P
NP
Art
N'
Leiter
aus
N
dem
Regal

5.



4.2

Gegeben seien folgende kontextfreie Sprachen:

$$L_1 = \left\{ ww^R | w \in \{a, b\}^+ \right\}$$
$$L_2 = \left\{ wcw^R | w \in \{a, b\}^* \right\}$$

(a) Intuitive PDAs

Ein PDA M ist als ein 6-Tupel $M = \langle K, \Sigma, \Gamma, \Delta, s, F \rangle$ definiert, wobei

K : Zustandsmenge
Σ : Eingabe-Alphabet (Zeichen des Eingabewortes)
Γ : Stack-Alphabet (Zeichen auf dem Stack)
Δ : Transitionsrelation ($\Delta \subseteq \langle \langle K, \Sigma^*, \Gamma^* \rangle, \langle K, \Gamma^* \rangle \rangle$)
s : Startzustand
F : Menge von Endzuständen

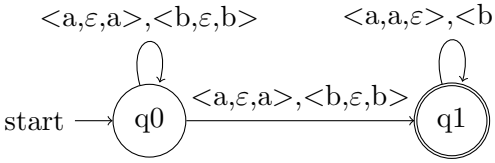
Die Bedingung, dass die Automaten dieser Teilaufgabe ohne Kategoriesymbole auf dem Stack auskommen, kann also als $\Gamma = \Sigma$ formuliert werden.

Im Folgenden werde ich der Einfachheit halber den Automaten mit einem Zustandsdiagramm angeben und lediglich Γ und Σ formal definieren.

$M_1 = \langle K_1, \Sigma_1, \Gamma_1, \Delta_1, s_1, F_1 \rangle$

$$\Sigma_1 = \{a, b\}$$
$$\Gamma_1 = \Sigma_1$$

$L(M_1) = L_1$

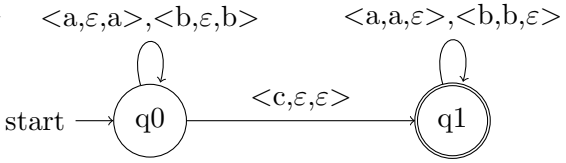


Zunächst werden die Symbole des halben Eingabewortes auf den Stack gelegt. Mit dem letzen Symbol wird nach q_1 gegangen. Dort werden Symbole vom Stack entfernt, wenn sie mit denen des Eingabewortes (zweite Hälfte) übereinstimmen.

$M_2 = \langle K_2, \Sigma_2, \Gamma_2, \Delta_2, s_2, F_2 \rangle$

$$\Sigma_2 = \{a, b, c\}$$
$$\Gamma_2 = \Sigma_2$$

$L(M_2) = L_2$



Zunächst werden die Symbole des halben Eingabewortes auf den Stack gelegt. Wird ein c im Eingabewort gelesen, wird nach q_1 gegangen. Dort werden Symbole von Stack entfernt, wenn sie mit denen des Eingabewortes (zweite Hälfte) übereinstimmen.

(b) Kontextfreie Grammatiken

Eine kontextfreie Grammatik (CFG) G ist definiert als ein 4-Tupel $G = \langle V, \Sigma, P, S \rangle$, wobei

V : Menge aller Symbole
Σ : Menge der Terminalsymbole ($\Sigma \subseteq V$)
P : Menge der Produktionsregeln (P $\subseteq (V \setminus \Sigma) \times V^*$)
S : Startsymbol $\in V \setminus \Sigma$

Der Lesbarkeit halber, gebe ich die Produktionsregeln in Pfeilschreibweise an.

$G_1 = \langle V_1, \Sigma_1, P_1, S_1 \rangle$

$$\Sigma_1 = \{a, b\}$$
$$V_1 = \Sigma_1 \cup \{S, S'\}$$
$$S_1 = S$$

Produktionsregeln P_1:

$$S \to aS'a$$
$$S \to bS'b$$
$$S' \to aS'a$$
$$S' \to bS'b$$
$$S' \to \varepsilon$$

$G_2 = \langle V_2, \Sigma_2, P_2, S_2 \rangle$

$$\Sigma_2 = \{a, b, c\}$$
$$V_2 = \Sigma_2 \cup \{S\}$$
$$S_2 = S$$

Produktionsregeln P_2:

$$S \to aSa$$
$$S \to bSb$$
$$S \to c$$

(c) Top-Down- bzw. Bottom-Up-Parser

Auch hier verwende ich wieder die Definition via Zustandsdiagramm.

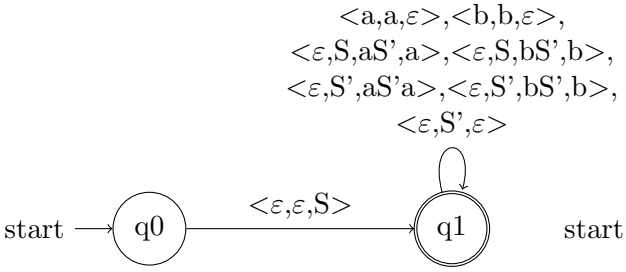
Top-Down-Parser:

$M_1' = \langle K_1', \Sigma_1', \Gamma_1', s_1', F_1' \rangle$

$\Sigma_1' = \Sigma_1$

$\Gamma_1' = \Sigma_1' \cup \{S, S'\}$

$L(M_1') = L_1$

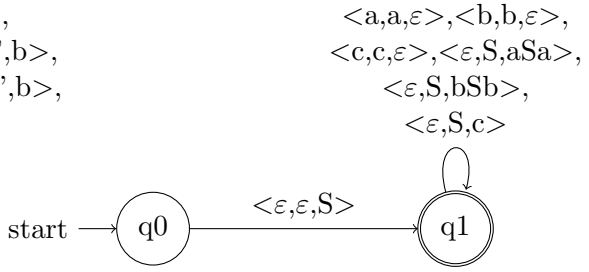


$M_2' = \langle K_2', \Sigma_2', \Gamma_2', s_2', F_2' \rangle$

$\Sigma_2' = \Sigma_2$

$\Gamma_2' = \Sigma_2' \cup \{S\}$

$L(M_2') = L_2$



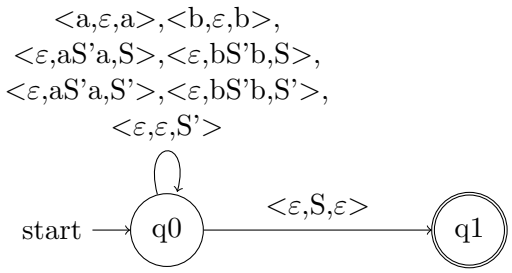
Bottom-Up-Parser (Shift-Reduce-Parser):

$M_1'' = \langle K_1'', \Sigma_1'', \Gamma_1'', \Delta_1'', s_1'', F_1'' \rangle$ mit

$\Sigma_1'' = \Sigma_1'$

$\Gamma_1'' = \Gamma_1'$

$L(M_1'') = L_1$

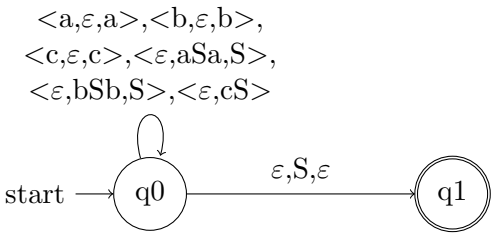


$M_2'' = \langle K_2'', \Sigma_2'', \Gamma_2'', \Delta_2'', s_2'', F_2'' \rangle$ mit

$\Sigma_2'' = \Sigma_2'$

$\Gamma_2'' = \Gamma_2'$

$L(M_2'') = L_2$



(d) Deterministische Automaten

Automat	deterministisch?
M_1	✗
M_2	✓
M_1'	✗
M_2'	✗
M_1''	✗
M_2''	✗

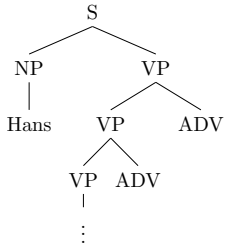
4.3

Gegeben sei die Grammatik mit den folgenden Produktionsregeln:

①	S	→	NP VP
②	VP	→	VP ADV
③	VP	→	geht
④	NP	→	Hans
⑤	ADV	→	heute
⑥	ADV	→	schnell

(a) Probleme beim Top-Down-Parsing

Da Regel ② der Grammatik linksrekursiv ist, kommt es beim Top-Down-Parsing zu einer Endlosschleife: Da die Regeln immer von oben nach unten abgearbeitet werden, wird eine VP immer zu VP ADV und die darin befindliche VP wieder zu VP ADV expandiert.



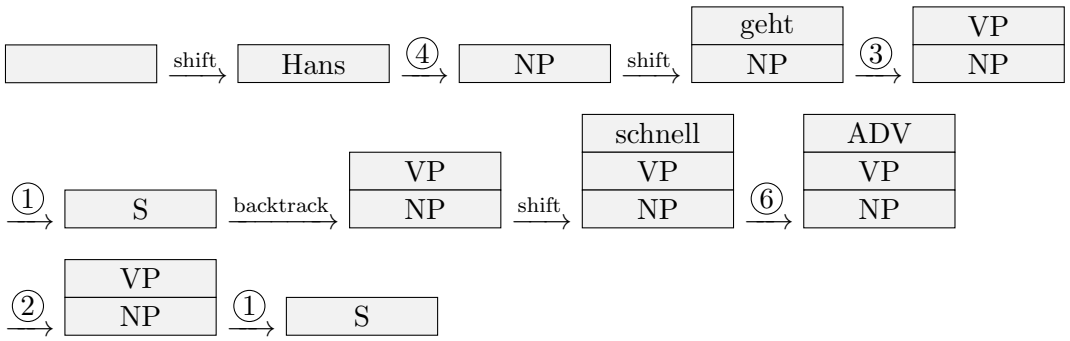
Die einfachste Lösung besteht darin, Regeln ② und ③ zu tauschen, sodass die Produktionsregeln in folgender Reihenfolge stehen:

①	S	→	NP VP
③	VP	→	geht
②	VP	→	VP ADV
④	NP	→	Hans
⑤	ADV	→	heute
⑥	ADV	→	schnell

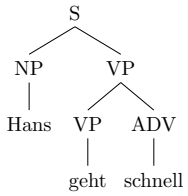
Dadurch wird verhindert, dass der Parser in eine endlose Rekursion geht.

(b) Shift-Reduce-Parsing

Im Folgenden wird der Durchlauf eines Shift-Reduce-Parsers beim Ableiten des Satzes *Hans geht schnell* aus der oben genannten ursprünglichen Grammatik angegeben.



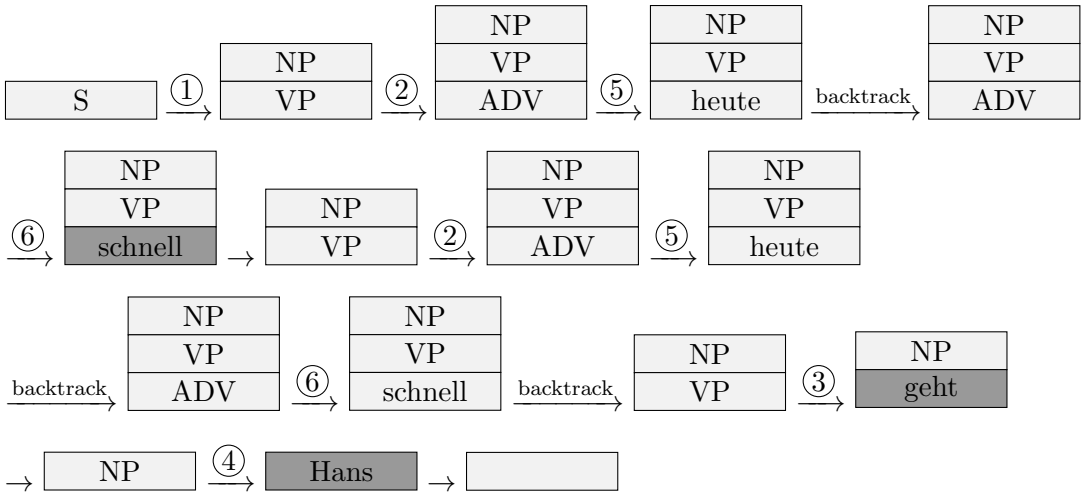
Folgender Syntax-Baum wird generiert:



(c) Top-Down-Parsing

Ein Top-Down-Parse war auf dem Übungsblatt nicht gefordert. Aber so sähe er aus, falls das von Interesse ist:

Im Folgenden wird ein Top-Down-Parser angenommen, der immer das am weitesten rechts stehende Nichtterminal expandiert. Dieser Parser wird, wie in Teilaufgabe (b), verwendet, um den Satz *Hans geht schnell* abzuleiten. Dunkle Zellen stehen für Terminale, die mit dem aktuellen Eingabewort übereinstimmen und daher im nächsten Schritt vom Stack entfernt werden können.



Der von diesem Parser erzeugte Syntaxbaum entspricht dem von Teilaufgabe (b).