Automatentheorie und Formale Sprachen

Vorlesung

Di 11:45 – 13:15 Fo 2 **Fr 10:45 – 11:30 Gr**	Beginn 20.4. (2h)	Prof. Dr. K. Indermark Sprechstunde n. Vereinb.

Zentralübung

Fr 10:00 – 10:45 Gr	Beginn 27.4.	B. Bollig Sprechstunde n. Vereinb.

Diskussionsgruppen (Beginn: 23.4.)

Mo 08:15 - 09:45 SG 12,
Mo 10:00 - 11:30 SG 413,
Mo 10:00 - 11:30 SG 512,
Mo 11:45 - 13:15 SG 413,
Mo 11:45 - 13:15 **AH III** (23.4., 30.4., 7.5.), **5052** (ab 14.5.)
Gruppenleiter: Patrick Hütten, Philipp Lucas,
Carsten Pelikan, Björn Riedel, Nico Wallmeier

Übungsschein

- 50% der erreichbaren Punkte bei den Übungsaufgaben
- Übungsscheinklausur (Mi, 18.7.2001, 15:00 – 17:00 Uhr)
- Abgabe der Lösungen maximal zu zweit

Aufgaben

- Ausgabe: freitags vor der Zentralübung (1. Blatt: 20.4.)
- Abgabe: freitags vor der Zentralübung (1. Blatt: 27.4.)

Vordiplomsklausur

- Informatik III, Teilprüfung ATFS
- Fr, 17.8.2001, Aula 2, AH IV, AH V, 9:00 – 12:00 Uhr
- Rechtzeitige Anmeldung im ZPA!

Web-Seite zur Vorlesung:

www-i2.informatik.rwth-aachen.de/Vorlesung/ATFS01/

- Folien
- Übungsblätter
- Literaturliste
- Aktuelles

Fortsetzung der Vorlesung

Berechenbarkeit und Komplexität (WS 2000/01), Prof. Thomas

- Algorithmenbegriff und Turingmaschinen
- WHILE- und GOTO-Programme
- Unentscheidbarkeit
- Zeit- und Platzkomplexität

Inhalt der Vorlesung

- Endliche Automaten und reguläre Sprachen
- Kellerautomaten und kontextfreie Sprachen
- Turingmaschinen und aufzählbare Sprachen

Literatur

- U. Schöning, *Theoretische Informatik kurz gefaßt*, Spektrum-Akademischer Verlag, 3. Auflage, 1997
- I. Wegener, *Theoretische Informatik*, Teubner, 2. Auflage, 1999
- K.W. Wagner, *Einführung in die Theoretische Informatik*, Springer Verlag, 1994
- J.E. Hopcroft, R. Motwani, J.D. Ullmann, *Introduction to Automata Theory, Languages, and Computation*, 2nd ed., Addison-Wesley, 2001
- J. v. Leeuwen, *Handbook of Theoretical Computer Science*, 2 Bände, Elsevier, 1990

Die griechische Wurzel des Wortes „Theorie“ heißt „sehen“.
Gut ausgewählte, verständlich dargestellte Theorie dient uns als
Lupe und Fernrohr, um Tatbestände zu sehen, die von
bloßem Auge unsichtbar sind.

aus: Nievergelt, Informatik-Spektrum 18 (1995), 342-344

Alphabete

$$
\begin{array}{lcl}
\Sigma_{\text{bin}} & = & \{0,1\} \\
\Sigma_{\text{dec}} & = & \{0,1,\ldots,9\} \\
\Sigma_{\text{hex}} & = & \Sigma_{\text{dec}} \cup \{\text{A,B,C,D,E,F}\} \\
\Sigma_{\text{let}} & = & \{\text{a,b},\ldots,\text{z,A,B},\ldots,\text{Z}\} \\
\Sigma_{\text{ascii}} & = & \text{ASCII-Zeichensatz}\ \ \text{(7-Bit)} \\
\Sigma_{\text{latin1}} & = & \text{Latin1-Zeichensatz (enthält ñ, Ñ) (8-Bit)} \\
\Sigma_{\text{unicode}} & = & \{\texttt{\textbackslash u}h_1h_2h_3h_4 \mid h_i \in \Sigma_{\text{hex}}\}\ \text{(Java) (16-Bit)} \\
\Sigma_{\text{sym}} & = & \{\ldots,<,=,>,<=,:=,->,\texttt{if},\texttt{while},\ldots\}
\end{array}
$$

Inhalt der Vorlesung

effektive Beschreibung formaler Sprachen durch

- reguläre Ausdrücke (RegE)
- nichtdeterministische endliche Automaten (NFA)
- deterministische endliche Automaten (DFA)

- kontextfreie Grammatiken (CFG)
- nichtdeterministische Kellerautomaten (PDA)
- deterministische Kellerautomaten (DPDA)

- Chomsky-Grammatiken vom Typ i (CG_i) für $i = 0, 1, 2, 3$
- nichtdeterministische Turingmaschinen (TM)
- deterministische Turingmaschinen (DTM)

- linear beschränkte Automaten (LBA)

Chomsky-Hierarchie

Für eine Sprachbeschreibung D sei $L(D)$ die beschriebene Sprache.

Für eine Beschreibungsklasse Δ sei

$$\mathcal{L}(\Sigma, \Delta) := \{L \subseteq \Sigma^* \mid \exists D \in \Delta : \; L = L(D)\}$$

die entsprechende Sprachklasse.

$$\begin{array}{ll} & \mathcal{L}(\Sigma, \mathrm{CG}_3) = \mathcal{L}(\Sigma, \mathrm{RegE}) = \mathcal{L}(\Sigma, \mathrm{NFA}) \; = \mathcal{L}(\Sigma, \mathrm{DFA}) \\ \subsetneq & \mathcal{L}(\Sigma, \mathrm{CG}_2) = \mathcal{L}(\Sigma, \mathrm{PDA}) \\ \subsetneq & \mathcal{L}(\Sigma, \mathrm{CG}_1) = \mathcal{L}(\Sigma, \mathrm{LBA}) \\ \subsetneq & \mathcal{L}(\Sigma, \mathrm{CG}_0) = \mathcal{L}(\Sigma, \mathrm{TM}) \quad = \mathcal{L}(\Sigma, \mathrm{DTM}) \end{array}$$

Grundprobleme von Beschreibungen

Sei Δ eine Beschreibungsklasse von Σ-Sprachen.

1) Das **Wortproblem** für Δ (WP(Δ)):
   Gibt es ein Verfahren, welches für $w \in \Sigma$ und $D \in \Delta$ feststellt, ob $w \in L(D)$ oder $w \notin L(D)$?

2) Das **Äquivalenzproblem** für Δ (ÄP(Δ)):
   Gibt es ein Verfahren, welches für $D_1, D_2 \in \Delta$ feststellt, ob $L(D_1) = L(D_2)$ oder $L(D_1) \neq L(D_2)$?

   Bez.: $D_1 \sim D_2$ („äquivalent“)$:\Leftrightarrow L(D_1) = L(D_2)$

3) Das **Leerheitsproblem** für Δ (LP(Δ)):
   Gibt es ein Verfahren, welches für $D \in \Delta$ feststellt, ob $L(D) = \emptyset$ oder $L(D) \neq \emptyset$?

Entscheidbarkeit dieser Probleme für Chomsky-Grammatiken

	WP	LP	ÄP
CG$_3$	+	+	+
CG$_2$	+	+	−
CG$_1$	+	−	−
CG$_0$	−	−	−

Inhalt

- Beschreibung formaler Sprachen durch reguläre Ausdrücke
- Erkennung durch nicht-det. und det. endliche Automaten
- Synthese und Analyse endlicher Automaten
- Zustandsreduktion endlicher Automaten
- Wortproblem, Äquivalenzproblem, Leerheitsproblem
- Anwendungen

UNIX-Notation für reguläre Ausdrücke

$\Sigma =$ASCII-Zeichensatz

$\texttt{.} \quad := \quad a_1 \vee \ldots \vee a_n \text{ mit } \{a_1, \ldots, a_n\} = \Sigma$

$\texttt{[abc]} \quad := \quad \texttt{a} \vee \texttt{b} \vee \texttt{c}$

$\texttt{[0-9]} \quad := \quad \texttt{0} \vee \texttt{1} \vee \ldots \vee \texttt{9}$

$\texttt{[A-Za-z0-9]} \quad := \quad \texttt{A} \vee \ldots \vee \texttt{Z} \vee \texttt{a} \vee \ldots \vee \texttt{z} \vee \texttt{0} \vee \ldots \vee \texttt{9}$

$\texttt{[:digit:]} \quad := \quad \texttt{[0-9]}$

$\texttt{[:alpha:]} \quad := \quad \texttt{[A-Za-z]}$

$\texttt{()} \quad := \quad \Lambda^*$

$\alpha\beta \quad := \quad \alpha \cdot \beta$

$\alpha\texttt{|}\beta \quad := \quad \alpha \vee \beta$

$\alpha\texttt{*} \quad := \quad \alpha^*$

$\alpha\texttt{+} \quad := \quad \alpha\alpha^*$

$\alpha\texttt{?} \quad := \quad \Lambda^*\texttt{|}\alpha$

$\alpha\{\texttt{3}\} \quad := \quad \alpha\alpha\alpha$

$\alpha\{3,5\} \quad := \quad \alpha\{\texttt{3}\}\texttt{|}\alpha\{\texttt{4}\}\texttt{|}\alpha\{\texttt{5}\}$

Ausführliche Beschreibung im Manual unter

```
> man 7 regex
```

(reg. expr. as defined in POSIX 1003.2)

Anwendungen

1) Symbolklassen von PS: Bezeichner,Zahlwörter,...
   Fließkommazahlen in C:
   ```
   ExpPart     = [eE][-+]?[0-9]+
   FractConst  = ([0-9]*\.[0-9]+)|([0-9]+\.)
   FloatConst  = ((({FractConst})({ExpPart})?)|
                  ([0-9]+{ExpPart}))[FfLl]?
   ```

2) Textsuche:
   ```
   > egrep 'Gr(ae|ä)del' publications.txt
   ```
   Ausgabe von Zeilen, die einen Match enthalten
   Besonderheit: $\hat{}\alpha$ Match von α am Zeilenanfang
   $\alpha\$$ Match von α am Zeilenende

3) Transformation von Texten mit Perl
   (Systemverwaltung)

4) Suchmaschinen für Web-Dokumente

Standardprobleme für RegE(Σ)

1) **Wortproblem** (matching problem):
   Sei $w \in \Sigma^*$ und $\alpha \in$ RegE(Σ). Ist w ein *Match* von α, also $w \in [\![\alpha]\!]$?
   Oder allgemeiner: Enthält w einen Match von α, d.h. ist w zerlegbar in $w = u_1 v u_2$ mit $v \in [\![\alpha]\!]$?

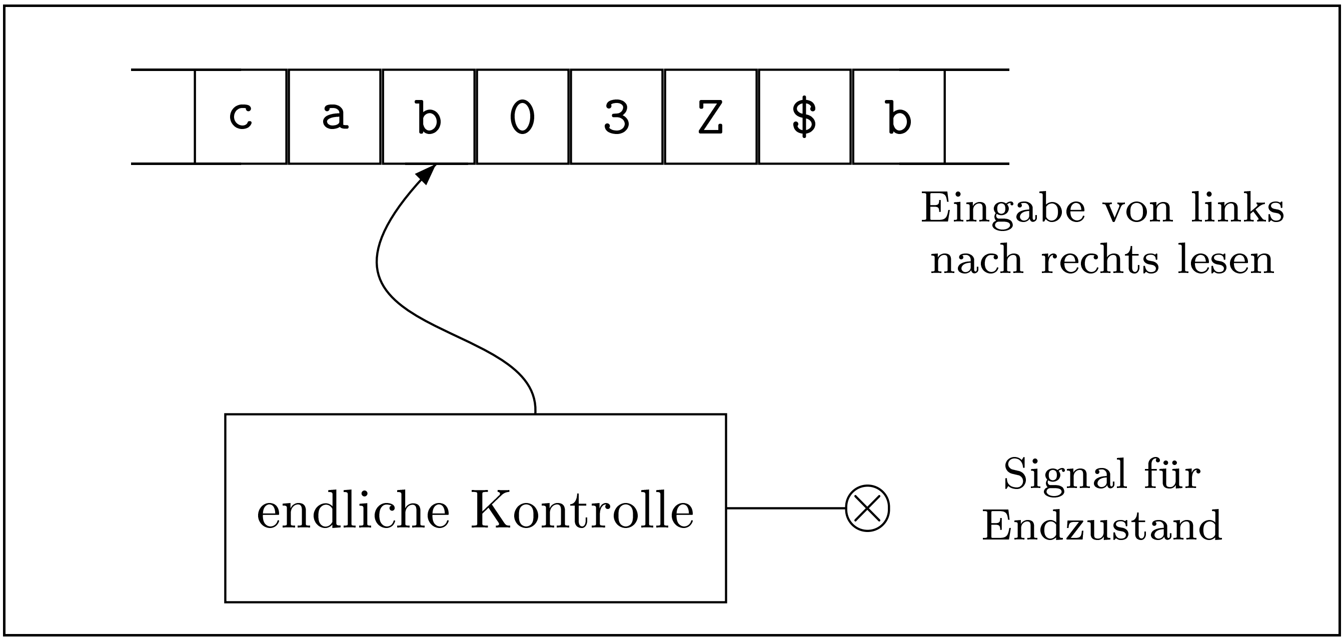
2) **Äquivalenzproblem**:
   Gilt $[\![\alpha]\!] = [\![\beta]\!]$ für $\alpha, \beta \in$ RegE(Σ)?
   Schreibweise: $\alpha \sim \beta :\Leftrightarrow [\![\alpha]\!] = [\![\beta]\!]$
   Beispiel: $\alpha^* \sim \Lambda^* \vee \alpha\alpha^*$

3) **Leerheitsproblem**:
   Gilt $[\![\alpha]\!] = \emptyset$ für $\alpha \in$ RegE(Σ)?

Darstellung eines DFA

durch

- Transitionstafel,
- Zustandsgraphen,
- Einwegleseband mit endlicher Kontrolle



Beispiel für einen DFA

Sei der DFA $\mathfrak{A} = \langle Q, \Sigma, \delta, q_0, F \rangle$ gegeben durch

$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3\}, \\ \Sigma &= \{a, b\}, \\ F &= \{q_3\} \quad \text{und} \end{aligned}$$

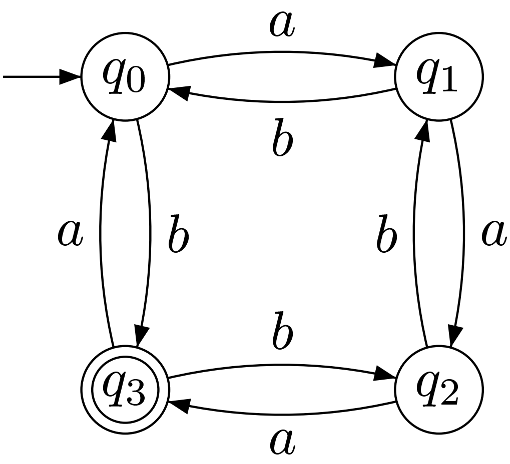
$$\begin{aligned} \delta(q_0, a) &= q_1, & \delta(q_0, b) &= q_3, \\ \delta(q_1, a) &= q_2, & \delta(q_1, b) &= q_0, \\ \delta(q_2, a) &= q_3, & \delta(q_2, b) &= q_1, \\ \delta(q_3, a) &= q_0, & \delta(q_3, b) &= q_2. \end{aligned}$$

Darstellung von $\mathfrak{A}$ durch

seine Transitionstafel:

δ	a	b
$\rightarrow q_0$	q_1	q_3
q_1	q_2	q_0
q_2	q_3	q_1
$\Rightarrow q_3$	q_0	q_2

seinen Zustandsgraphen:



Es gilt: $L(\mathfrak{A}) = \{w \in \Sigma^* \mid |w|_a - |w|_b \equiv 3 \bmod 4\}$

Beispiel für einen NFA (1)

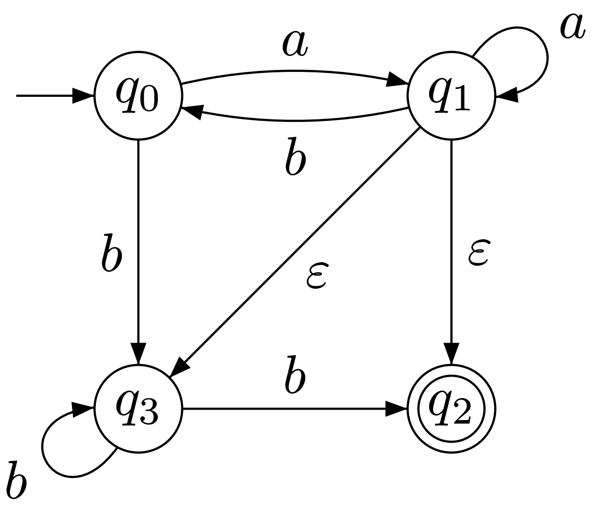
Sei der NFA

$$\mathfrak{A} = \langle Q, \Sigma, \delta, q_0, F \rangle$$

gegeben durch seine Transitionstafel

δ	ε	a	b
$\rightarrow q_0$	$\emptyset$	$\{q_1\}$	$\{q_3\}$
q_1	$\{q_2, q_3\}$	$\{q_1\}$	$\{q_0\}$
$\Rightarrow q_2$	$\emptyset$	$\emptyset$	$\emptyset$
q_3	$\emptyset$	$\emptyset$	$\{q_2, q_3\}$

bzw. seinen Zustandsgraphen



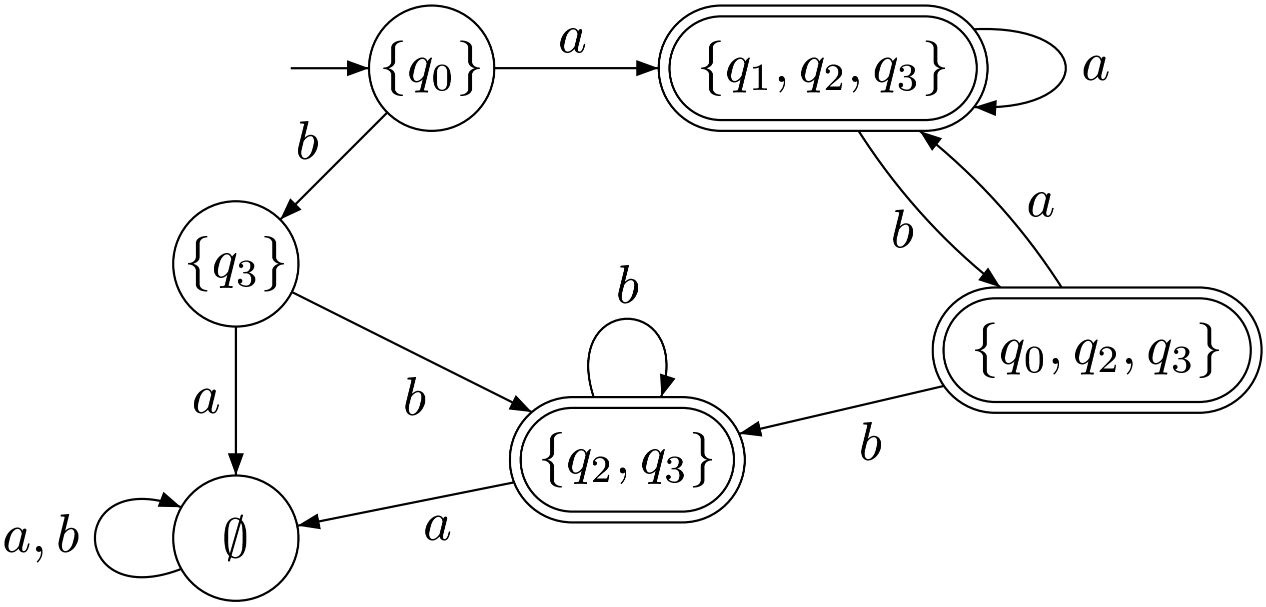
Es gilt: $L(\mathfrak{A}) = [\![(aa^*b)^*bb^*b \vee a(a \vee ba)^*(b^*b \vee \Lambda^*)]\!]$

Beispiel für einen NFA (2)

Der Potenzmengenautomat $\mathfrak{A}^P = \langle \widehat{Q}, \Sigma, \widehat{\delta}, \widehat{q_0}, \widehat{F} \rangle$ von $\mathfrak{A}$ ist wie folgt durch seine Transitionstafel gegeben:

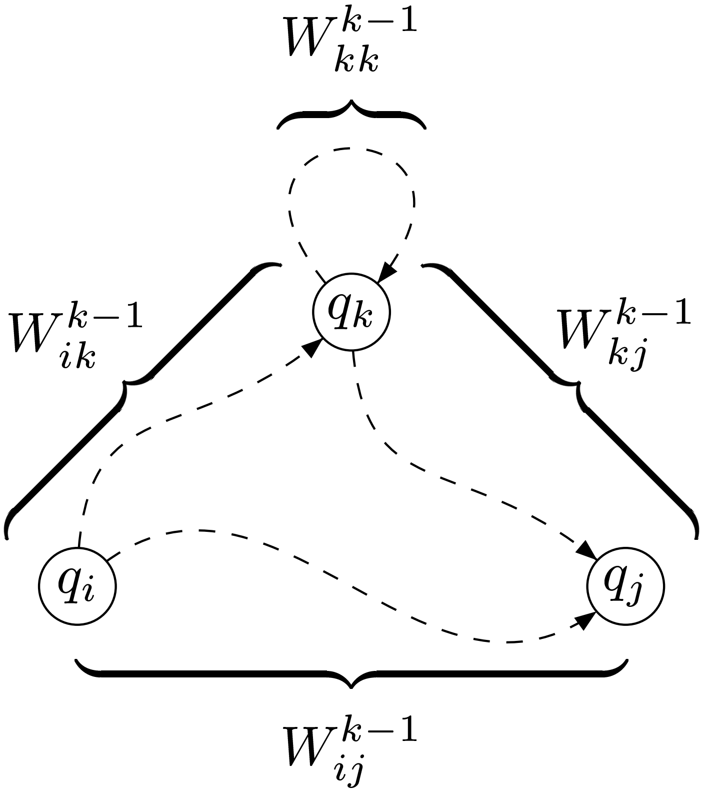
$\widehat{\delta}$	a	b
$\rightarrow \{q_0\}$	$\{q_1, q_2, q_3\}$	$\{q_3\}$
$\{q_3\}$	$\emptyset$	$\{q_2, q_3\}$
$\emptyset$	$\emptyset$	$\emptyset$
$\Rightarrow \{q_2, q_3\}$	$\emptyset$	$\{q_2, q_3\}$
$\Rightarrow \{q_1, q_2, q_3\}$	$\{q_1, q_2, q_3\}$	$\{q_0, q_2, q_3\}$
$\Rightarrow \{q_0, q_2, q_3\}$	$\{q_1, q_2, q_3\}$	$\{q_2, q_3\}$

und wie folgt durch seinen Zustandsgraphen:



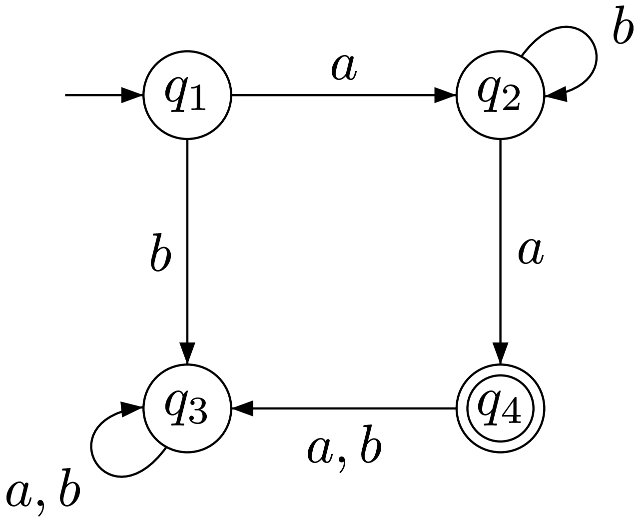
W_{ij}^{k}-Mengen

$$\begin{array}{rrcl} & W_{ij}^{0} & \subseteq & \Sigma \cup \{\varepsilon\} \\ k > 0: & W_{ij}^{k} & = & W_{ij}^{k-1} \;\cup\; W_{ik}^{k-1}\,(W_{kk}^{k-1})^{*}\,W_{kj}^{k-1} \end{array}$$



Gleichungssystem eines DFA

DFA $\mathfrak{A}$ mit $L(\mathfrak{A}) = [\![ab^*a]\!]$:



Gleichungssystem von Sprachen:

$$\begin{array}{lclclclcl}
X_1 & = & \{a\}X_2 & \cup & \{b\}X_3 & & & & \\
X_2 & = & \{b\}X_2 & \cup & & & \{a\}X_4 & & \\
X_3 & = & & & \{a,b\}X_3 & & & & \\
X_4 & = & & & \{a,b\}X_3 & \cup & & & \{\varepsilon\}
\end{array}$$

Äquivalenzensystem von regulären Ausdrücken:

$$\begin{array}{lclclclcl}
x_1 & \sim & a\ x_2 & \vee & b\ x_3 & & & & \\
x_2 & \sim & b\ x_2 & \vee & & & a\ x_4 & & \\
x_3 & \sim & & & (a \vee b)\ x_3 & & & & \\
x_4 & \sim & & & (a \vee b)\ x_3 & \vee & & & \Lambda^*
\end{array}$$

Resolutionsregel

Für $\alpha, \beta, \gamma \in \mathrm{RegE}(\Sigma)$ gilt:
Wenn $\alpha \sim \beta\alpha \vee \gamma$ und $\varepsilon \notin [\![\beta]\!]$, so $\alpha \sim \beta^*\gamma$.

Beweis: Seien $A, B, C \subseteq \Sigma^*$. Wir zeigen die entsprechende Behauptung:

$$A = BA \cup C,\ \varepsilon \notin B \curvearrowright A = B^*C$$

Da für alle $n > 0$

$$A = B^nA \cup B^{n-1}C \cup B^{n-2}C \cup \ldots \cup BC \cup C,$$

folgt $B^*C \subseteq A$.

Andererseits:
$w \in A,\ |w| = k,\ \varepsilon \notin B \curvearrowright$
$w \notin B^{k+1}A \curvearrowright$
$w \in B^kC \cup B^{k-1}C \cup \ldots \cup BC \cup C \subseteq B^*C$

Lösung eines Äquivalenzensystems

Äquivalenzensystem (siehe Folie 2.10):

$$\begin{array}{lllllll} x_1 & \sim & a\ x_2 & \vee & b\ x_3 & & \\ x_2 & \sim & b\ x_2 & \vee & & & a\ x_4 \\ x_3 & \sim & & & (a \vee b)\ x_3 & & \\ x_4 & \sim & & & (a \vee b)\ x_3 & \vee & \Lambda^* \end{array}$$

Lösung:

$$\begin{array}{llllll} & x_3 & \sim & (a \vee b)^* \Lambda & \sim & \Lambda \\ \rightsquigarrow & x_4 & \sim & \Lambda^* & & \\ \rightsquigarrow & x_2 & \sim & b^* a \Lambda^* & \sim & b^* a \\ \rightsquigarrow & x_1 & \sim & a b^* a & & \end{array}$$

Markierungsalgorithmus

DFA :

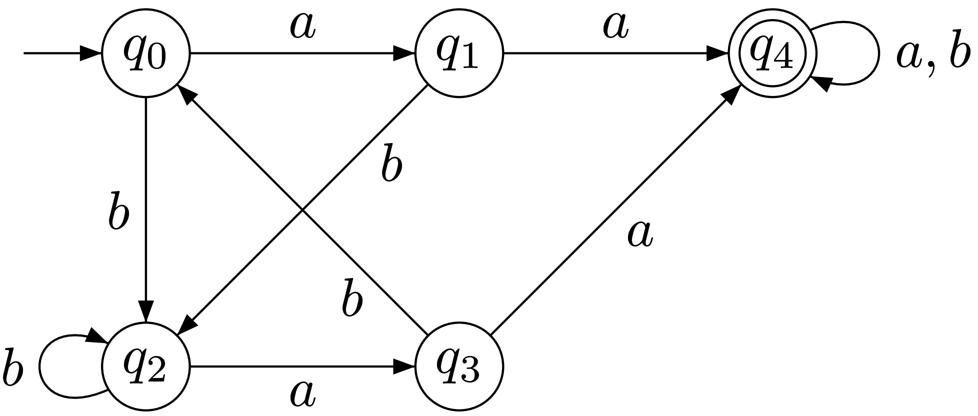
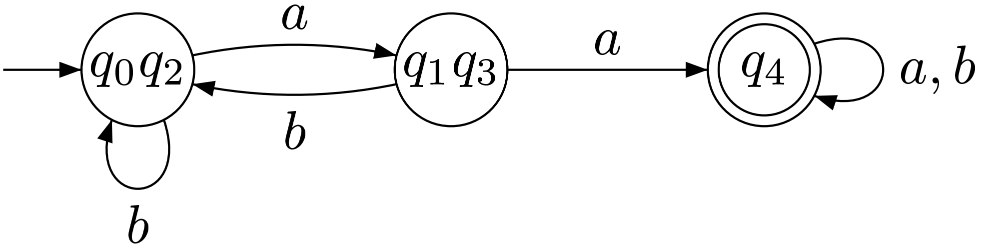


Tabelle der Zustandspaare:

q_1				
q_2				
q_3				
q_4				
	q_0	q_1	q_2	q_3

Minimaler DFA :



Anwendung 1: Mealy-Automat

$$\mathfrak{A} = \langle Q, \Sigma, \Delta, q_0, \delta, \lambda \rangle$$

mit Δ Ausgabealphabet

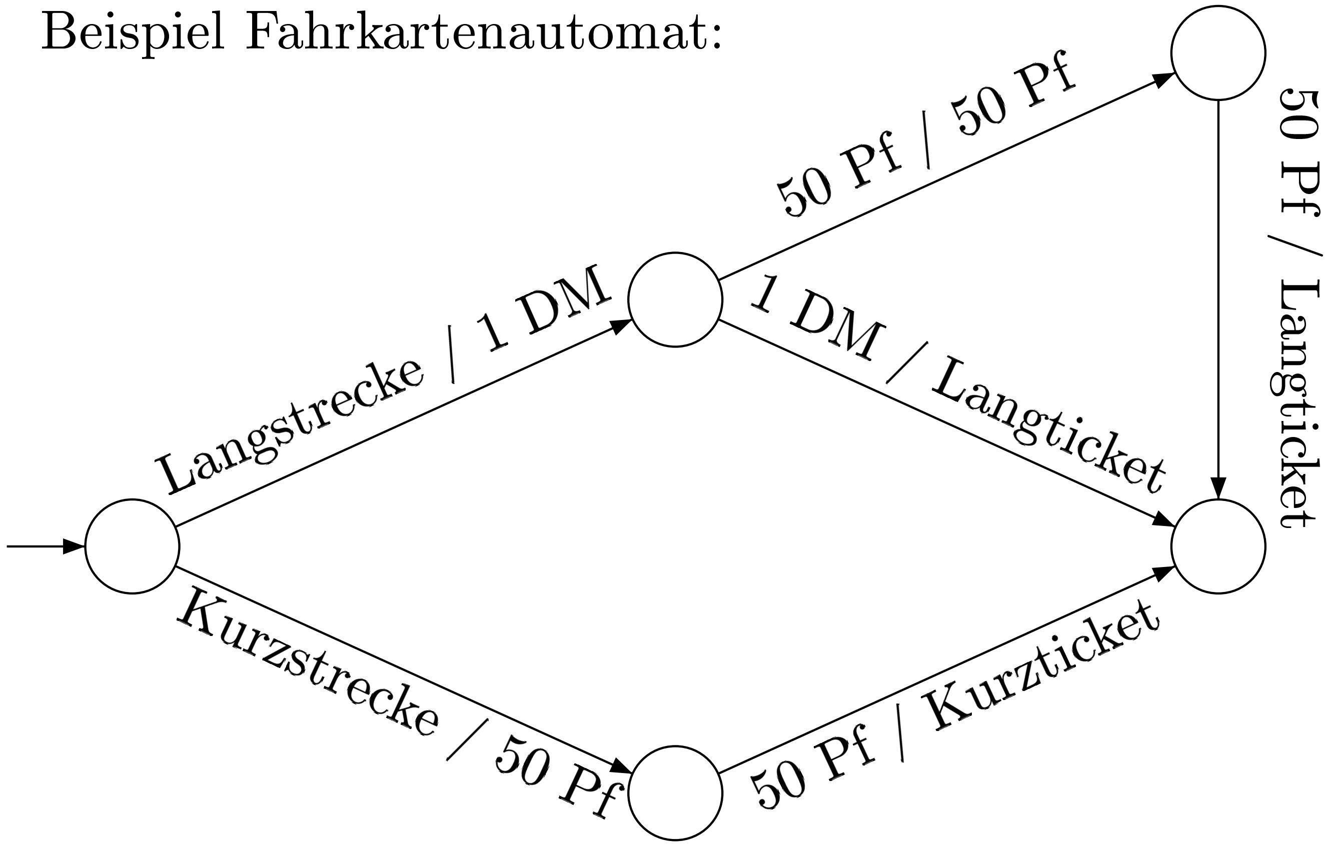
$\lambda : Q \times \Sigma \to \Delta$ Ausgabefunktion

$f_{\mathfrak{A}} : \Sigma^* \longrightarrow \Delta^*$ Berechnungsfunktion

$f_{\mathfrak{A}}(\varepsilon) = \varepsilon$

$f_{\mathfrak{A}}(wa) = f_{\mathfrak{A}}(w)\lambda(\overline{\delta}(q_0, w), a)$

Beispiel Fahrkartenautomat:



Anwendung 2: Wortsuche in Texten

Der Suchautomat von {web,ebay}:

siehe Hopcroft, Motwani, Ullmann, Seite 70

Implementierungstechniken
a) NFA-Methode
b) DFA-Methode:

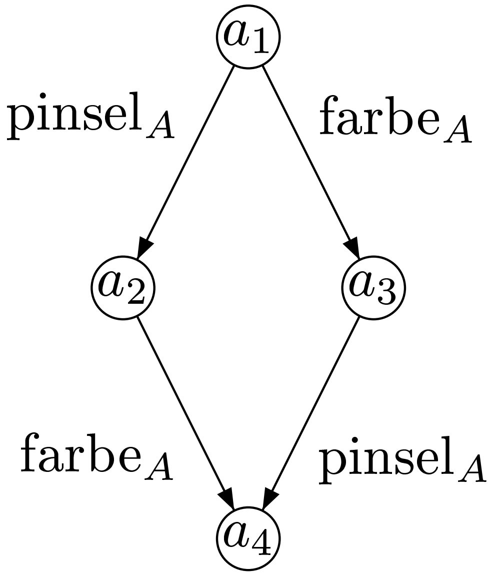
siehe Hopcroft, Motwani, Ullmann, Seite 71

Anwendung 3: Theorie paralleler Prozesse (1)

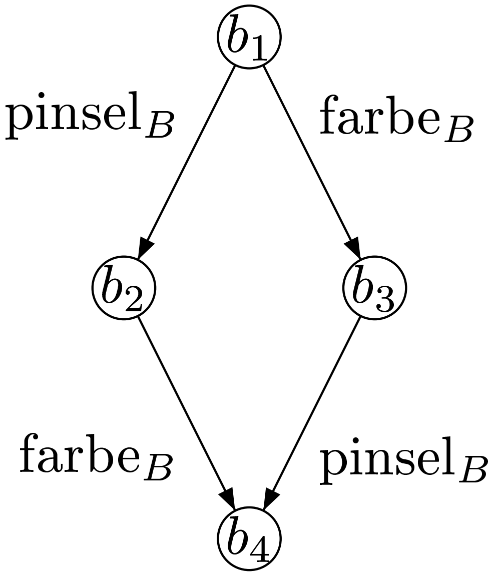
Beispiel:

- Anna und Bernd wollen malen
- benötigen dazu Pinsel und Farbe

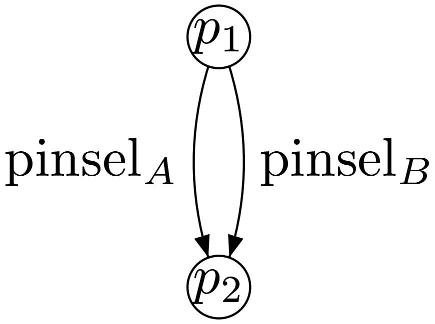
Anna:



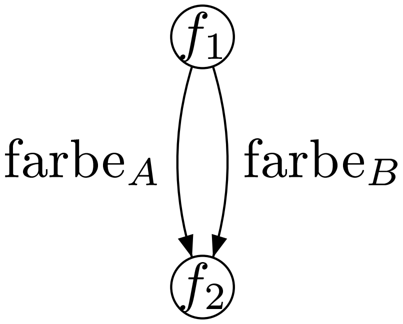
Bernd:



Pinsel:

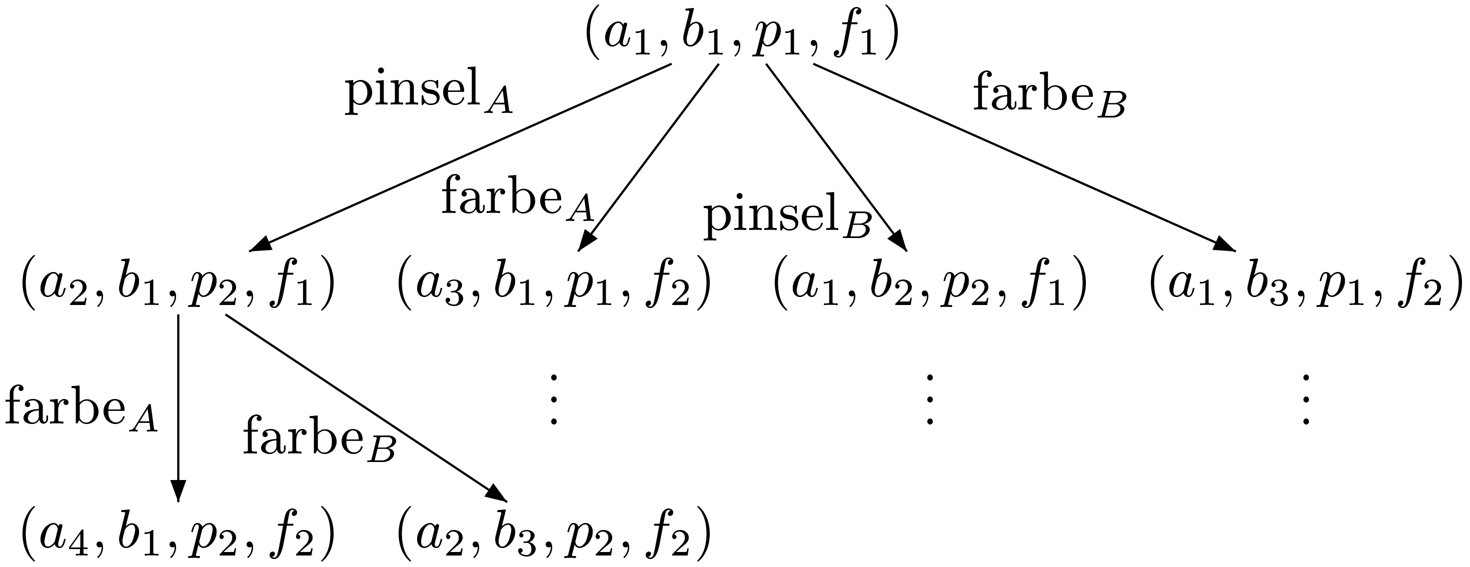


Farbe:



Anwendung 3: Theorie paralleler Prozesse (2)

Transitionsdiagramm:



(a_1, b_1, p_1, f_1)

pinsel_A farbe_A pinsel_B farbe_B

(a_2, b_1, p_2, f_1) (a_3, b_1, p_1, f_2) (a_1, b_2, p_2, f_1) (a_1, b_3, p_1, f_2)

farbe_A farbe_B $\vdots$ $\vdots$ $\vdots$

(a_4, b_1, p_2, f_2) (a_2, b_3, p_2, f_2)

“Deadlock”

Zusammenfassung

- Beschreibung formaler Sprachen durch reguläre Ausdrücke
- Erkennung durch nicht-det. und det. endliche Automaten
- Synthese und Analyse endlicher Automaten
- Pumping-Lemma
- Zustandsminimierung endlicher Automaten
- entscheidbare Eigenschaften endlicher Automaten
  - Wortproblem
  - Äquivalenzproblem
  - Leerheitsproblem
- Anwendungen
  - Automaten mit Ausgabe
  - Wortsuche in Texten
  - Theorie paralleler Prozesse

Inhalt

- kontextfreie Grammatiken (CFG), Eindeutigkeit
- einseitig-lineare Grammatiken
- Normalformen von CFG: CNF, GNF
- Abschlußeigenschaften von CFL, Pumping-Lemma
- entscheidbare Eigenschaften von CFG
- Kellerautomaten
- CYK-Algorithmus
- erweiterte CFG (EBNF)
- rekursive NFA / Syntaxdiagramm

Historie

1) Noam Chomsky (1955):
“Phrasenstrukturgrammatik”
Beispiel:
<Satz> $\longrightarrow$ <Nominalphrase> <Verbalphrase>
<Nominalphrase> $\longrightarrow$ <Artikel> <Nomen>
<Verbalphrase> $\longrightarrow$ <Verb> <Nominalphrase>
<Artikel> $\longrightarrow$ die
<Nomen> $\longrightarrow$ Katze
<Verb> $\longrightarrow$ frißt
<Nomen> $\longrightarrow$ Maus

<Satz> $\Longrightarrow^*$ die Katze frißt die Maus

2) John Backus (1959): “Backus-Naur-Form” (BNF)
Beispiel:
<exp> ::= <exp> <op> <exp>
<exp> ::= (<exp>)
<exp> ::= $-$<exp>
<exp> ::= id
<op> ::= $+ \mid - \mid * \mid / \mid \uparrow$

<exp> $\Longrightarrow^*$ $\mathrm{id} + (\mathrm{id} - (\mathrm{id} \uparrow \mathrm{id}))$

Kontextfreie Grammatik von WHILE-Programmen

$$\langle \{S, A, B\}, \Sigma, P, S \rangle$$

$$\Sigma = \{\text{begin}, ;, \text{end}, \text{if}, \text{then}, \text{else}, \text{while}, \text{do}, :=, +1, -1, > 0\} \cup \{X_i \mid i \in \mathbb{N}\}$$

$$\begin{array}{ll} P: & S \longrightarrow A \mid \text{begin } S;\ S \text{ end} \mid \text{if } B \text{ then } S \text{ else } S \mid \text{while } B \text{ do } S \\ & A \longrightarrow X_i := X_i + 1 \mid X_i := X_i - 1 \quad i \in \mathbb{N} \\ & B \longrightarrow X_i > 0 \quad i \in \mathbb{N} \end{array}$$

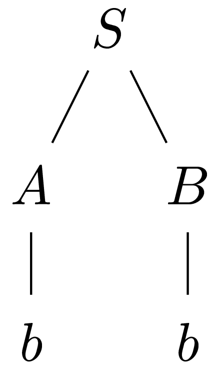
$$\begin{array}{lll} S & \Longrightarrow & \text{while } B \text{ do } S \\ & \Longrightarrow & \text{while } X_1 > 0 \text{ do } S \\ & \Longrightarrow & \text{while } X_1 > 0 \text{ do begin } S;\ S \text{ end} \\ & \Longrightarrow & \text{while } X_1 > 0 \text{ do begin } X_1 := X_1 - 1;\ S \text{ end} \\ & \Longrightarrow & \text{while } X_1 > 0 \text{ do begin } X_1 := X_1 - 1;\ X_0 := X_0 + 1 \text{ end} \end{array}$$

Rechts- und Linksableitung

Sei die kontextfreie Grammatik G gegeben durch

$$\begin{array}{lcl} S & \longrightarrow & AB \\ A & \longrightarrow & Aa \mid b \\ B & \longrightarrow & ab \mid b \end{array}$$

Der Ableitungsbaum



entspricht den (verschiedenen) Ableitungen

$$S \Longrightarrow AB \Longrightarrow bB \Longrightarrow bb$$

$$S \Longrightarrow AB \Longrightarrow Ab \Longrightarrow bb$$

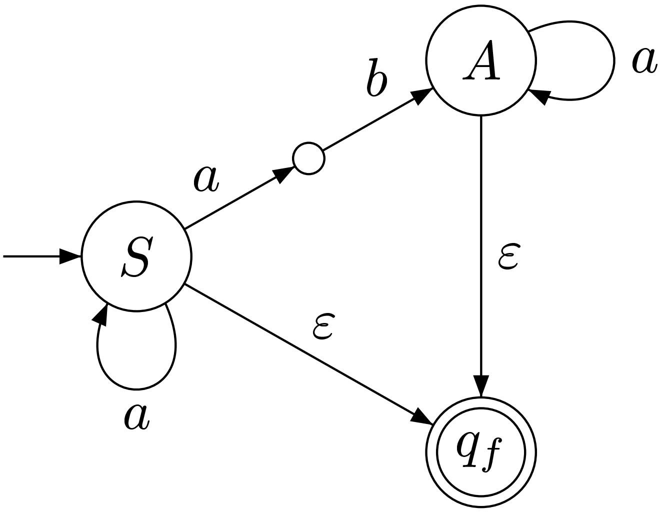
($\leadsto$ Links- und Rechtsableitungen)

Rechtslineare Grammatiken und endliche Automaten

Sei die rechtslineare Grammatik G gegeben durch

$$\begin{array}{lcl} S & \longrightarrow & aS \mid abA \mid \varepsilon \\ A & \longrightarrow & aA \mid \varepsilon \end{array}$$

$\mathfrak{A}(G)$ ist gegeben durch



Es gilt $L(G) = L(\mathfrak{A}(G)) = [\![a^*(\Lambda^* \vee aba^*)]\!]$.

Zusammenfassung

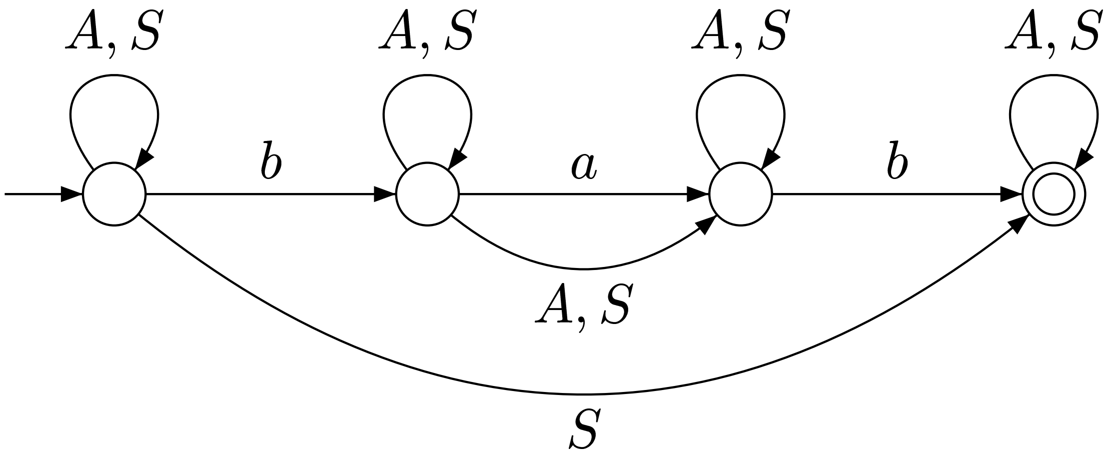
$$\begin{array}{rl} & \{L \subseteq \Sigma^* \mid L = L(G),\ G \text{ rechtslinear}\} \\ = & \{L \subseteq \Sigma^* \mid L = L(G),\ G \text{ linkslinear}\} \\ = & \{L \subseteq \Sigma^* \mid L = L(G),\ G \text{ einseitig linear}\} \\ = & \mathcal{L}(\Sigma, \mathrm{NFA}) \\ = & \mathcal{L}(\Sigma, \mathrm{DFA}) \\ = & [\![\mathrm{RegE}(\Sigma)]\!] \\ = & \{L \subseteq \Sigma^* \mid |D(L)| < \infty\} \\ = & \mathrm{RegL}(\Sigma) \end{array}$$

Vorgängerabschluß

Sei die Grammatik G gegeben durch

$$\begin{array}{lcl} S & \longrightarrow & bSb \mid A \\ A & \longrightarrow & aA \mid \varepsilon \end{array}$$

Konstruktion eines NFA, der $Pre^*_G(\{bab\})$ erkennt:



Kellerautomat (PDA)

Die Sprache der Palindrome mit markierter Mitte,

$$L = \{wcw^R \mid w \in \{a,b\}^*\},$$

wird erkannt vom PDA

$$\mathfrak{A} = \langle \{q_0, q_1, q_2\}, \{a,b,c\}, \{a,b,Z_0\}, \delta, q_0, Z_0, \{q_2\} \rangle$$

mit den Transitionen (δ):

$$\begin{array}{ccc l}
(q_0, x, A) & \longrightarrow & (q_0, xA) & \\
(q_0, c, A) & \longrightarrow & (q_1, A) & \\
(q_1, x, x) & \longrightarrow & (q_1, \varepsilon) & (\forall A \in \{a,b,Z_0\}, \\
(q_1, \varepsilon, Z_0) & \longrightarrow & (q_2, \varepsilon) & \;\forall x \in \{a,b\})
\end{array}$$

Berechnungsbeispiel:

	Zustand	Eingabe (Kopf links)	Keller (Spitze links)
	$(q_0,$	$aca,$	$Z_0)$
$\vdash$	$(q_0,$	$ca,$	$aZ_0)$
$\vdash$	$(q_1,$	$a,$	$aZ_0)$
$\vdash$	$(q_1,$	,	$Z_0)$
$\vdash$	$(q_2,$	,	)

Beispiel zu $\mathbf{CFL}(\Sigma) \subseteq \mathfrak{L}(\Sigma, \mathbf{PDA})$

Sei $G \in \mathrm{CFG}(\{a, +, *\})$ gegeben durch

$$S \longrightarrow S + a \mid S * a \mid a$$

Dann ist

$\mathfrak{A}_G = \langle \{q\}, \{+, *, a\}, \{+, *, a, S\}, \delta, q, S \rangle \in \mathrm{PDA}(\{a, +, *\})$,

wobei δ wie folgt gegeben ist:

$$\begin{array}{llll}
(1) & (q, +, +) \longrightarrow (q, \varepsilon) & (4) & (q, \varepsilon, S) \longrightarrow (q, S + a) \\
(2) & (q, *, *) \longrightarrow (q, \varepsilon) & (5) & (q, \varepsilon, S) \longrightarrow (q, S * a) \\
(3) & (q, a, a) \longrightarrow (q, \varepsilon) & (6) & (q, \varepsilon, S) \longrightarrow (q, a)
\end{array}$$

$$a * a + a \in L(G) = L(\mathfrak{A}_G, \varepsilon):$$

Linksableitung in G: Erkennung durch $\mathfrak{A}_G$:

$$\begin{array}{ll}
 & S \\
\Longrightarrow & S + a \\
\Longrightarrow & S * a + a \\
\Longrightarrow & a * a + a
\end{array}$$

$$\begin{array}{lrrr}
 & (q, & a * a + a, & S) \\
\vdash^{(4)} & (q, & a * a + a, & S + a) \\
\vdash^{(5)} & (q, & a * a + a, & S * a + a) \\
\vdash^{(6)} & (q, & a * a + a, & a * a + a) \\
\vdash^{(3)} & (q, & *a + a, & *a + a) \\
\vdash^{(2)} & (q, & a + a, & a + a) \\
\vdash^{(3)} & (q, & +a, & +a) \\
\vdash^{(1)} & (q, & a, & a) \\
\vdash^{(3)} & (q, & \varepsilon, & \varepsilon)
\end{array}$$

CYK-Algorithmus

Eingabe: $G \in \mathrm{CNF}, w = a_1 \ldots a_n, n > 0$

Ausgabe: N_{ij} für alle i, j mit $1 \leq i \leq j \leq n$

Verfahren:

```
for $i := 1$ to $n$ do
   $N_{i,i} := \{A \mid A \to a_i \in P\}$
for $d := 1$ to $n-1$ do     (* $d = j - i$ *)
   for $i := 1$ to $n-d$ do     (* $N_{i,i+d}$-Berechnung *)
      begin
         $N_{i,i+d} := \emptyset$
         for $k := i$ to $i + (d-1)$ do
            $N_{i,i+d} := N_{i,i+d} \cup \{A \mid A \to BC \in P,$
                                            $B \in N_{i,k}, C \in N_{k+1,i+d}\}$
      end
```

Beispiel zum CYK-Algorithmus

Sei die Grammatik G gegeben durch

$$\begin{array}{lcl} S & \longrightarrow & AB \mid a \\ A & \longrightarrow & BA \mid a \\ B & \longrightarrow & AB \mid b \end{array}$$

Dann gilt:

$$S \Longrightarrow AB \Longrightarrow^2 BAAB \Longrightarrow^4 baab$$

Tabelle der N_{ij}:

$i \setminus j$	1	2	3	4
1	$\{B\}$	$\{A\}$	$\emptyset$	$\{S, B\}$
2		$\{S, A\}$	$\emptyset$	$\{S, B\}$
3			$\{S, A\}$	$\{S, B\}$
4				$\{B\}$
	b	a	a	b

Extended-Backus-Naur-Form und erweiterte kontextfreie Grammatiken

EBNF:

$$\begin{aligned} \mathit{Statement} &::= \mathit{Var} := \mathit{Expr} \\ &\quad\mid \text{if } \mathit{Expr} \text{ then } \mathit{Statement} \text{ [else } \mathit{Statement}] \\ &\quad\mid \text{begin } \mathit{Statement} \ \{;\ \mathit{Statement}\} \text{ end} \\ &\quad\mid \text{repeat } \mathit{Statement} \ \{;\ \mathit{Statement}\} \text{ until } \mathit{Expr} \\ \mathit{Expr} &::= \mathit{Var} \mid \text{true} \mid \text{false} \\ \mathit{Var} &::= \text{x} \mid \text{y} \mid \text{z} \end{aligned}$$

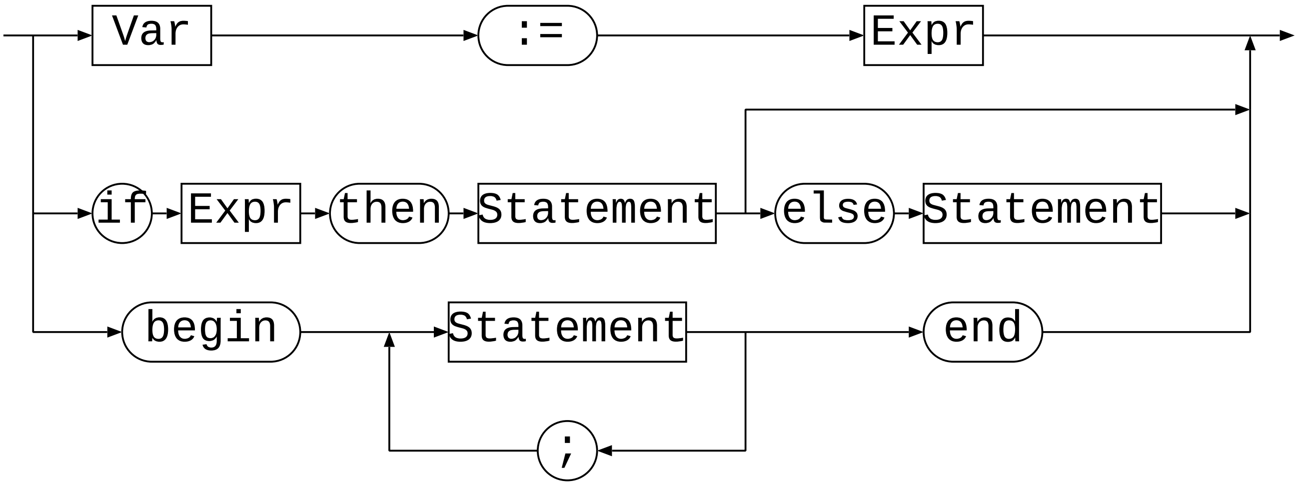
ECFG:

$$\begin{array}{ccl} S & \longrightarrow & V := E \\ & \vee & \text{if } E \text{ then } S \text{ (else } S \vee \Lambda^*) \\ & \vee & \text{begin } S\ (;\ S)^* \text{ end} \\ & \vee & \text{repeat } S\ (;\ S)^* \text{ until } E \\ E & \longrightarrow & V \vee \text{true} \vee \text{false} \\ V & \longrightarrow & \text{x} \vee \text{y} \vee \text{z} \end{array}$$

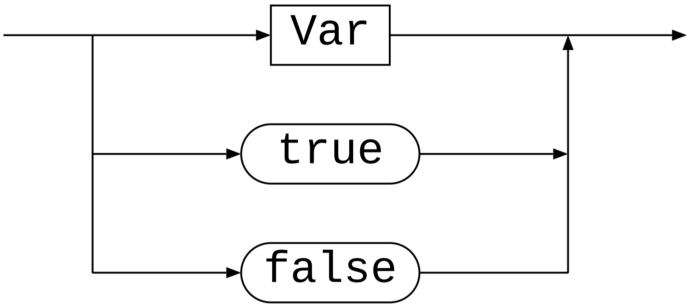
$$\Sigma = \{\text{x}, \text{y}, \text{z}, \text{true}, \text{false}, :=, \ldots\}$$

Syntaxdiagramm

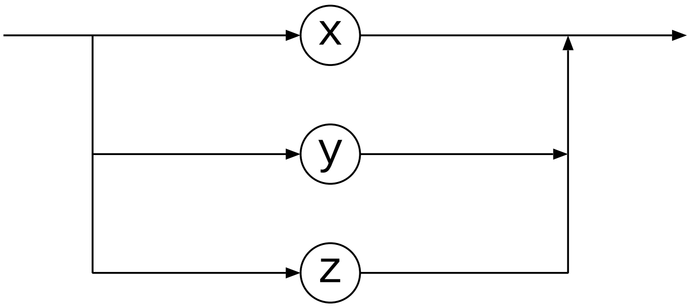
Statement:



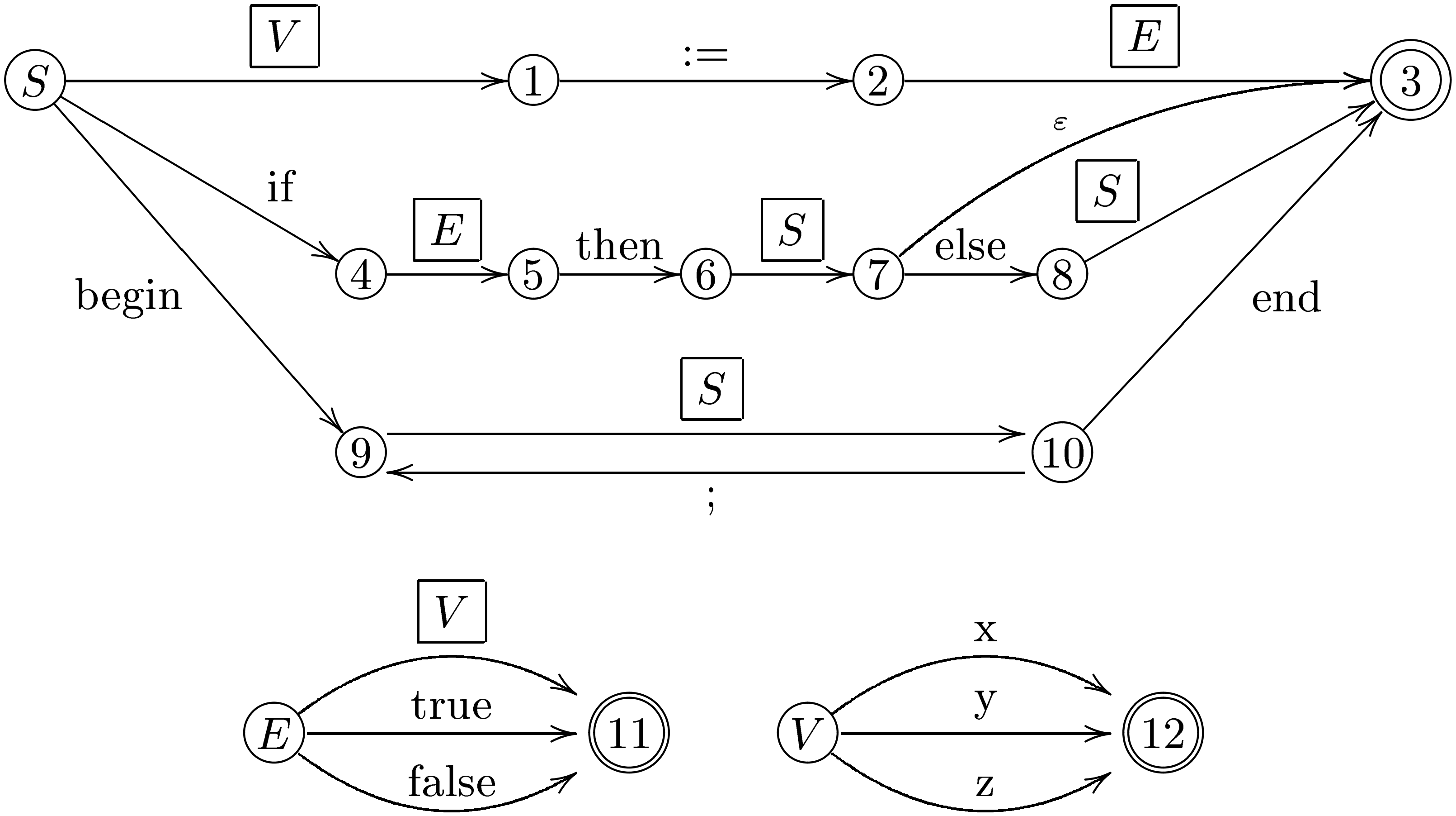
Expr:



Var:



Rekursiver endlicher Automat



Erkennende Berechnung des rekursiven endlichen Automaten

$$\begin{array}{cl}
 & (S, \text{if true then x := false}) \\
\vdash & (4, \text{true then x := false}) \\
\overset{(1)}{\vdash} & (5, \text{then x := false}) \\
\vdash & (6, \text{x := false}) \\
\overset{(2)}{\vdash} & (7, \varepsilon) \\
\vdash & (3, \varepsilon)
\end{array}$$

(1) wegen:

$$(E, \text{true then x := false}) \quad \vdash \quad (11, \text{then x := false})$$

(2) wegen:

$$(S, \text{x := false}) \overset{(3)}{\vdash} (1, \text{:= false}) \vdash (2, \text{false}) \overset{(4)}{\vdash} (3, \varepsilon)$$

(3) wegen:

$$(V, \text{x := false}) \quad \vdash \quad (12, \text{:= false})$$

(4) wegen:

$$(E, \text{false}) \quad \vdash \quad (11, \varepsilon)$$

Inhalt

- Chomsky-Grammatiken und Chomsky-Hierarchie
- Abschluß von $\mathcal{L}_0$ unter regulären Operationen und Schnitt
- Platzbedarfssatz
- Abschluß von $\mathcal{L}_1$ unter regulären Operationen und Schnitt
- Turingmaschinen, linear beschränkte Automaten
- aufzählbare und entscheidbare Sprachen

Die Chomsky-Hierarchie

Für $|\Sigma| > 1$ gilt:

$$\begin{array}{ccccccccc}
\mathcal{L}_3(\Sigma) & \underset{\neq}{\subseteq} & \mathcal{L}_2(\Sigma) & \underset{\neq}{\subseteq} & \mathcal{L}_1(\Sigma) & \underset{\neq}{\subseteq} & \mathrm{DEC}(\Sigma) & \underset{\neq}{\subseteq} & \mathcal{L}_0(\Sigma) \\
\| & & \| & & \| & & & & \| \\
\mathrm{RegL}(\Sigma) & & \mathcal{L}(\Sigma, \mathrm{PDA}) & & \mathcal{L}(\Sigma, \mathrm{LBA}) & & & & \mathcal{L}(\Sigma, \mathrm{TM}) \\
\| & & \cup\!\nparallel & & & & & & \| \\
\mathcal{L}(\Sigma, \mathrm{DFA}) & & \mathcal{L}(\Sigma, \mathrm{DPDA}) & & & & & & \mathcal{L}(\Sigma, \mathrm{DTM}) \\
\| & & & & & & & & \\
\mathcal{L}(\Sigma, \mathrm{NFA}) & & & & & & & &
\end{array}$$

Abschlußeigenschaften

	Schnitt	Vereinigung	Komplement	Produkt	Stern
$\mathcal{L}_3$	ja	ja	ja	ja	ja
$\mathcal{L}_2$	nein	ja	nein	ja	ja
$\mathcal{L}_1$	ja	ja	ja	ja	ja
$\mathcal{L}_0$	ja	ja	nein	ja	ja

Entscheidbarkeit

	Wortproblem	Leerheitsproblem	Äquivalenzproblem
$\mathcal{L}_3$	ja	ja	ja
$\mathcal{L}_2$	ja	ja	nein
$\mathcal{L}_1$	ja	nein	nein
$\mathcal{L}_0$	nein	nein	nein