



Automaten, Sprachen und Berechenbarkeit

Prof. Schlosser-Haupt
SS 2009

Inhaltsverzeichnis

1	Einführung	3
1.1	Grundbegriffe	3
1.2	Motivation	4
1.3	Grammatiken	4

1 Einführung

1.1 Grundbegriffe

Alphabet: $\Sigma = \{a_1, \dots, a_k\} \neq \emptyset$ mit lexikographischer Ordnung $a_i < a_{i+1}$
 $1 \leq i \leq k-1$
 $i, k \in \mathbb{N} := \{1, 2, \dots\}$

Bsp: $\Sigma_1 = \{a, b, c\}$, $\Sigma_2 = \{0, 1\}$

Wort (über Σ): endliche Folge von Zeichen aus Σ (Zeichenkette, engl. string)

Bsp: *baa* Wort über Σ_1 , 0110 Wort über Σ_2 : "Binärwort"

Wir bezeichnen mit ε das leere Wort: $\varepsilon\omega = \omega\varepsilon = \omega \forall$ Worte ω und mit Σ^* die Menge aller Worte (über Σ) inklusive ε ; $\Sigma^+ = \Sigma^* - \{\varepsilon\}$

Bsp: $\Sigma_2^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, \dots\}$, angegeben in "lexikographischer" Ordnung

$|\omega|$: Länge des Wortes $\omega \hat{=}$ Anzahl der Zeichen

Hinweis: Σ^* ist für alle Σ abzählbar unendlich.

$$f: \mathbb{N}_0 \longrightarrow \Sigma^* = \{f(0), f(1), \dots\}, f \text{ bijektiv}$$

Bsp: $\Sigma_2^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, \dots\}$. In diesem Fall kann man das f sogar konkret angeben!

$$f(0) = \varepsilon, f(1) = 0, f(2) = 1, f(3) = 00, f(4) = 01 \text{ usw.}$$

Das ist für "abzählbar unendlich" nicht gefordert.

Eine beliebige Teilmenge L von Σ^* wird als (formale) Sprache über Σ bezeichnet. Für $L_1, L_2 \subseteq \Sigma^*$ wird $L_1L_2 = \{v\omega : v \in L_1, \omega \in L_2\}$ als Konkatenation bezeichnet ("hintereinander geschrieben").

Bsp: $L_1 = \{a^n : n \in \mathbb{N}_0\}$, $L_2 = \{b^m : m \in \mathbb{N}_0\}$ sind Sprachen über Σ_1 , $L_1L_2 = \{a^n b^m : n, m \in \mathbb{N}_0\}$, wobei $a^0 = \varepsilon, a^n = aa^{n-1}, n \in \mathbb{N}$.

Für $L \subseteq \Sigma^*$ setzt man $L^0 = \{\varepsilon\}$ - die Sprache ist also nicht die leere Sprache $\emptyset = \{\}$ - und $L^n = LL^{n-1}, n \in \mathbb{N}$

$L^* = \bigcup_{n \geq 0} L^n, L^+ = L^* - \{\varepsilon\} = \bigcup_{n \geq 1} L^n$. L^* heißt Kleenesche Hülle oder Kleeneabschluß von L ,

L^+ positive Hülle, $P(L) = \{S : S \subseteq L\}$ Potenzmenge (von L). (Kleene: 1909 - 1998)

Bsp: $L = \{a, ab\}, \Sigma = \{a, b\}$

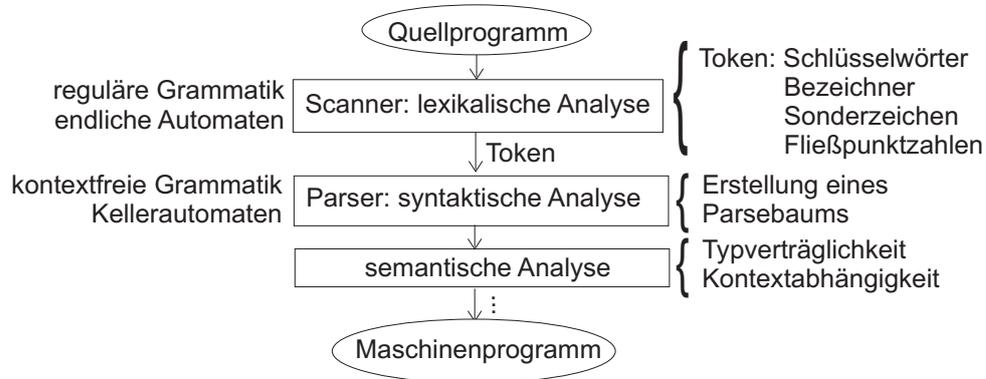
$$L^1 = LL^0 = L\{\varepsilon\} = L$$

$$L^2 = LL^1 = \{a, ab\}\{a, ab\} = \{a^2, a^2b, aba, (ab)^2\}$$

$$P(L) = \{\emptyset, \{a\}, \{ab\}, L\}$$

1.2 Motivation

Formale Sprachen/Automatentheorie spielen eine wichtige Rolle in Schaltkreistheorie, Textverarbeitung, Compiler, ...



Merke: Grammatiken definieren Sprachen, Automaten überprüfen, ob das vorgelegte Wort (z.B. das Programm) den Syntaxregeln der betreffenden Grammatiken entspricht: "Wortproblem"

1.3 Grammatiken

Bsp 1: $\Sigma = \{A, B, \dots, Z, a, \dots, z, 0, \dots, 9\}$

$$L_{Bu} = \{A, \dots, Z, a, \dots, z\}$$

$$L_{Zi} = \{0, \dots, 9\}$$

$$L_{Bez} = L_{Bu}(L_{Bu} \cup L_{Zi})^*$$

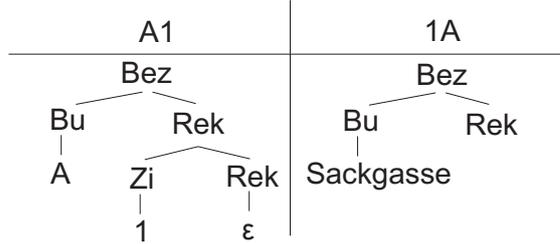
$$P_{Bez} = \begin{cases} Bu \longrightarrow A|B|\dots|z \\ Zi \longrightarrow 0|1|\dots|9 \\ Bez \longrightarrow Bu \text{ Rek} \\ Rek \longrightarrow Bu \text{ Rek}|Zi \text{ Rek}|\varepsilon \end{cases}$$

P_{Bez} ist eine Zusammenfassung der Ersetzungsregeln, dabei steht P für Produktion/Regel. Die Variablen auf der linken Seite nennt man Nichtterminale; die Menge sei \overline{V} . Die Zeichen in Σ nennt man in diesem Zusammenhang Terminalsymbole, weil sie sich nicht weiter ersetzen lassen.

1 Einführung

Wortproblem

$A1 \in L_{Bez}$ bzw. $1A \in L_{Bez}$?



$A1 \in L_{Bez}$

$1A \notin L_{Bez}$, weil das Wort nicht den Regeln entspricht

Die Variablen, die die Wurzel des Ableitungsbaums bildet - hier Bez - und sich auf der linken Seite der Startregel befindet, nennt man Startsymbol \underline{S} .

Zusammenfassung: $G_{Bez} = (\Sigma, V, P_{Bez}, S)$
 $V = (Bu, Zi, Bez, Rek)$
 $S = Bez$
 $L_{Bez} := L(G_{Bez})$

Der amerikanische Sprachwissenschaftler Noam Chomsky führte 1959 die folgenden Begriffe ein:

Def 2: Eine Grammatik G wird definiert durch:

\underline{V} Alphabet der Variablen

$\underline{\Sigma}$ Alphabet der Terminalsymbole

\underline{P} endliche Menge von Regeln

$[p \subseteq (V \cup \Sigma)^* V (V \cup \Sigma)^* \times (V \cup \Sigma)^* = (\varphi, \psi) \in P : \varphi \rightarrow \psi, \varphi \neq \varepsilon]$

$S \in V$ Startsymbol

$G = (V, \Sigma, P, S)$

Verabredung: Variablen: Großbuchstaben

Terminale: kleine Buchstaben, Alphabetanfang

Worte: kleine Buchstaben am Ende

Def 3: $G = (V, \Sigma, P, S)$, $x, y \in (V \cup \Sigma)^*$

$x \Rightarrow y$ (y in G ableitbar aus x) genau dann, wenn

$$\left. \begin{array}{l} x = \omega\varphi z \\ \Downarrow \\ y = \omega\psi z \end{array} \right\} \text{ mit Regel } \varphi \rightarrow \psi \text{ aus } P$$

$x \xRightarrow{*} y$ genau dann, wenn

$x = x_0 \Rightarrow x_1 \Rightarrow \dots \Rightarrow x_n = y$

1 Einführung

Def 4: a) $G = (V, \Sigma, P, S)$. Die von der Grammatik G erzeugte/definierte Sprache
 $L(G) = \{\omega \in \Sigma^* : S^* \Rightarrow \omega\}$

b) G_1 und G_2 über demselben Terminalalphabet Σ heißen äquivalent ($G_1 \sim G_2$),
wenn $L(G_1) = L(G_2)$

zu Bsp 1: "Konstruktion einer äquivalenten Grammit G'_{Bez} "

$$P'_{Bez} = \begin{cases} Bez \longrightarrow Bu\ Rek|Bu \\ Rek \longrightarrow Bu\ Rek|Zi\ Rek|Bu|Zi \\ Bu \longrightarrow A|B|\dots|z \\ Zi \longrightarrow 0|\dots|9 \end{cases}$$

$G'_{Bez} = (V, \Sigma, P'_{Bez}, Bez)$ ist " ϵ -frei"

$G'_{Bez} \sim G_{Bez}$

Bsp 5: $G = (\{S\}, \{0, 1\}, P, S)$ mit

$$P = \{s \longrightarrow 0s1|01\}$$

$$L(G) = \{\omega \in \{0, 1\}^* : \omega = 0^n 1^n \text{ für ein } n \in \mathbb{N}\}$$

ist die von G erzeugte Sprache.