- Grundbegriffe
  - Wort
    - Präfix
      - wenn es $z \in \sum^*$ gibt mit $yz = w$
    - Infix/Faktor
      - wenn es $x, z \in \sum^*$ gibt mit $xyz = w$
    - Suffix
      - wenn es $x \in \sum^*$ gibt mit $xy = w$
  - Sprache
    - Chomsky Hierachie
      - Typ 0: Allgemein
jede Grammatik ist vom Typ 0
      - Typ 1: Kontextsensitiv
wenn es Wörter $u, v, w \in (V \cup \sum)^*, |v| > 0$ und ein Nichtterminal $A \in V$ gibt mit $l = uAw$ und $r = uvw$
      - Typ 2: Kontextfrei
wenn $l \in V$ und $r \in (V \cup \sum)^*$ gilt
      - Typ 3: Regulär
wenn $l \in V$ und $r \in \sum V \cup \epsilon$ gilt
    - Kleene Abschluss $L* = \bigcup_{n \geq 0} L^n$
  - Grammatik
    - Symbole
      - Nicht-Terminale, Großbuchstaben, Elemente aus V
      - Terminale, Kleinbuchstaben, Elemente aus $\sum$
    - 4-Tupel $G = (V, \sum, P, S)$
      - V ist eine endliche Menge von Nicht-Terminalen oder Variablen
      - $\sum$ ist Alphabet (Menge der Terminale)
      - P ist eine endliche Menge von Regeln oder Produktionen
      - $S \in V$ ist das Startsymbol oder das Axiom

- intuitiv (loop) berechenbar
  - μ rekurisv
    - Definition
      - $\mu f : \mathbb{N}^k \to \mathbb{N}$ definiert durch
      - $\mu f(n_1, ..., n_k) = min\{m | f(m, n_1, ..., n_k) = 0$ und
      - $\forall x < m : f(x, n_1, ..., n_k)$ definiert $\}$.
  - while berechnenbar
    - Definition
      - $x_i = c; x_i = x_j + c; x_i = x_j - c$ mit $c \in \{0, 1\}$ und $i, j \geq 1$ (Wertzuweisung) oder
      - $P_1; P_2$, wobei P_1 und P_2 bereits While Programme sind oder
      - while $x_i \neq 0$ do P end, wobei P ein While Programm ist und $i \geq 1$.
    - Gödel Vermutung
  - goto berechnenbar
    - endliche nichtleere File $P = A_1; A_2; ...; A_m$ von Anweisungen A_i der Form
    - $x_i = c, x_i = x_j + c, x_i = x_j - c$ mit $c \in \{0, 1\}$ und $i, j \geq 1$
    - goto l mit $0 \leq l \leq m$ (unbedingter Sprung)
    - if $x_i = 0$ then l mit $i \geq 1$ und $0 \leq l \leq m$ (bedingter Sprung)
  - Turing berechenbar
    - Berechnung
      - Eingabe x ist anfänglicher Bandinhalt
      - Ausgabe $f(x)$ ist Bandinhalt (ohne Blanks am Rand)beim Erreichen eines Endzustands
      - Hält die TM für die gegebene Eingabe nicht an, ist $f(x)$ an dieser Stelle undefiniert.
    - Church-Turing-These
      - Alle im intuitiven Sinne berechenbaren Funktionen können mit Turingmaschinen berechnet werden.

Rechtslineare Sprachen

- **endliche Automaten (Maschinen)**
  - DFA M
    - von einem DFA akzeptierte Sprache ist: $L(M) = w \in \sum^* |\hat{\delta}(z_0, w) \in E$
    - Eine Sprache $L \supseteq \sum^*$ ist regulär, wenn es einen DFA mit $L(M) = L$ gibt
  - NFA M
    - Jede von einem NFA akzeptierte Sprache ist regulär
  - Satz
    - Wenn L_1 und L_2 reguläre Sprachen sind, dann ist auch
    - $L_1 \cup L_2$ regulär
    - $L_1 \cap L_2$ regulär
    - $L_1 L_2$ regulär
    - L_1^+ / L_1^* regulär
  - Jede reguläre Sprache ist rechtslinear
- **Reguläre Ausdrücke**
  - Definition
    - Die Menge $Reg(\sum)$ der regulären Ausdrücke über dem Alphabet $\sum$ ist die kleinste Menge mit folgenden Eigenschaften:
    - $\varnothing \in Reg(\sum), \lambda \in Reg(\sum), \sum \subseteq Reg(\sum)$
    - Wenn $\alpha, \beta \in Reg(\sum)$, dann auch $(\alpha * \beta), (\alpha + \beta), (\alpha^*) \in Reg(\sum)$
    - für $\alpha * \beta$ schreibt man oft $\alpha\beta$
    - für $\alpha + \beta$ schreibt man auch $\alpha|\beta$
  - Für einen regulären Ausdruck $\alpha \in Reg(\sum)$ ist die Sprache $L(\alpha) \subseteq \sum^*$ induktiv definiert
    - zu jedem regulären Ausdruck γ gibt es einen NFA M mit $L(\gamma) = L(M)$
    - zu jedem DFA M gibt es einen regulären Ausdruck γ mit $L(M) = L(\gamma)$
- **Nicht-Reguläre Sprachen**
  - Pumping Lemma
    - L sei reguläre Sprache, dann gibt es $n \leq 1$ derart, dass für alle $x \in L$ mit $|x| \geq n$ gilt: es gibt Wörter $u, v, w \in \sum^*$ mit
    - $x = uvw$, $|uv| \leq n$, $|v| \geq 1$
    - $uv^i w \in L$ für alle $i \geq 0$
    - geeignet um Aussagen über Nicht-Regularität zu machen
  - Myhill-Nerode Äquivalenz
    - binäre Relation $R_L \subseteq \sum^* \times \sum^*$
    - $\forall x, y \in \sum^*$ setze $(x, y) \in R_L$ genau dann, wenn $\forall z \in \sum^*$: $(xy \in L \leftrightarrow yz \in L)$ gilt. xR_Ly
    - Für Sprache L und Wort $x \in \sum^*$ ist $[x]_L = \{y \in \sum^* |xR_Ly\}$ die Äquivalenzklasse von x
    - Satz: L ist regulär $\leftrightarrow$ $index(R_L) < \infty$
- **Entscheidbarkeit**
  - Wortproblem
    - Gilt $w \in L$ für eine gegebene reguläre Sprache L und $w \in \sum^*$?
  - Leerheitsproblem
    - Gilt $L = \varnothing$ für eine gegebene reguläre Sprache L?
  - Endlichkeitsproblem
    - Ist eine gegebene reguläre Sprache L endlich?
  - Schnittproblem
    - Gilt $L_1 \cap L_2 = \varnothing$ für gegebene reguläre L_1, L_2?
  - Inklusionsproblem
    - Gilt $L_1 \subseteq L_2$ für gegebene reguläre L_1, L_2?
  - Äquivalenzproblem
    - Gilt $L_1 = L_2$ für gegebene reguläre L_1, L_2?

deterministischer endlicher Automat M

- 5-Tupel $M = (Z, \sum, z_0, \delta, E)$
- Z eine endliche Menge von Zuständen
- $\sum$ das Eingabealphabet (mit $Z \cap \sum = \emptyset$)
- $z_0 \in Z$ der Startzustand
- $\delta : Z \times \sum \rightarrow Z$ die Übergangsfunktion
- $E \subseteq Z$ die Menge der Endzustände
- kurz: DFA (deterministic finite automaton)

Turingmaschine (TM)

- 7-Tupel $M = (Z, \sum, \Phi, \delta, z_o, \Box, E)$
- $\sum$ das Eingabealphabet
- Φ mit $\Phi \supseteq \sum$ und $\Phi \cap Z \neq 0$ das Arbeits- oder Bandalphabet,
- $z_0 \in Z$ der Startzustand,
- $\delta : Z \times \Phi \rightarrow (Z \times \Phi \times \{L, N, R\})$ die Überführungsfunktion
- $\Box \in \Phi / \sum$ das Leerzeichen oder Blank und
- $E \subseteq Z$ die Menge der Endzustände ist

Linksableitung

Eine Ableitung $S = w_0 \Rightarrow ... \Rightarrow w_n = w \in \sum^*$ heißt Linksableitung, wenn in jedem Schritt das am weitesten links stehende Nichtterminal ersetzt wird (analog Rechtsabl.)

CYK-Algorithmus

Gehört ein gegebenes Wort zu $L(G)$?

	a_1	a_2	$\cdots$	$\cdots$	a_{n-1}	a_n
$j = 1$	$V_{1,1}$	$V_{2,1}$	$\cdots$	$\cdots$	$V_{n-1,1}$	$V_{n,1}$
$j = 2$	$V_{1,2}$	$V_{2,2}$	$\cdots$	$\cdots$	$V_{n-1,2}$	
	$\cdots$	$\cdots$	$\cdots$	$\cdots$		
$\vdots$	$\cdots$	$\cdots$	$\cdots$			
$j = n-1$	$V_{1,n-1}$	$V_{2,n-1}$				
$j = n$	$V_{1,n}$					

Von oben nach unten, von links unten nach rechts oben

Kellerautomaten

Um ein Automatenmodell für Kontextfreie Sprachen zu erhalten führt man einen Keller-(Pushdown)-Speicher ein, auf dem sich eine beliebig lange Sequenz von Zeichen befinden darf. Beim Einlesen eines neuen Zeichens wird das oberste Zeichen des Kellers gelesen und durch eine (evtl. leere) Sequenz von Zeichen ersetzt. An anderen Stellen kann der Keller nicht gelesen/geändert werden

Greibach-Normalform

Eine kontextfreie Grammatik G ist in Greibach Normalform falls alle Produktionen aus P folgende Form haben: $A \rightarrow aB_1B_2...B_k$, mit $k \in \mathbb{N}, A, B_1, ..., B_k \in V$ und $a \in \sum$. Die Greibach Normalform garantiert, dass bei jedem Ableitungsschritt genau ein Alphabetsymbol entsteht.

Lemma von Ogden (William Ogden)

Wenn L eine kontextfreie Sprache ist, dann gibt es $n \geq 1$ derart, dass für alle $z \in L$, in denen n Positionen markiert sind, gilt: es gibt Wörter $u, v, w, x, y \in \sum^*$ mit

1. $z = uvwxy$
2. v oder x enthält wenigstens eine der Markierungen oder
3. $uv^iwx^iy \in L$ für alle $i \geq 0$

Halteproblem

allgemein: Das Halteproblem ist die Menge aller Paare (M, x),wobei M eine TM ist und $x \in \{0,1\}^*$, sodass M bei Eingabe von x hält.

Reduktion

Seien $A \subseteq \sum^*, B \subseteq \Phi^*$. Eine Reduktion von A auf B ist eine totale und berechenbare Funktion $f : \sum^* \rightarrow \Phi^*$, so dass für alle $w \in \sum^*$ gilt: $w \in A \leftrightarrow f(x) \in B$. A heißt auf B reduzierbar (in Zeichen $A \leq B$), falls es eine Reduktion von A auf B gibt.

Satz von Rice

Sei R die Klasse aller Turing-berechenbaren Funktionen $\{0,1\}^* \rightarrow \{0,1\}^*$, Ω die nirgendwo definierte Funktion und sei $S \subseteq \mathbb{R}$ mit $\Omega \in S$ und $\neq \mathbb{R}$. Dann ist die Sprache $C(S) = \{w \in L_{TM} | \phi_w \in S\}$ unentscheidbar.

Semi Entscheidbarkeit

Auch wenn das Halteproblem bei leerer Eingabe H_0 unentscheidbar ist, so kann doch nach endlicher Zeit festgestellt werden, daß die Maschine M_w bei leerer Eingabe anhält - H_0 ist also „halb-" oder „semi-entscheidbar".
Eine Sprache $L \subseteq \sum^*$ heißt semi-entscheidbar, falls die „halbe" charakteristische Funktion von L, d.h. die partielle Funktion $X'_L : \sum^* \rightarrow \{1\}$ mit $x'_L = \begin{cases} 1 & \text{falls } w \in L \\ undef. & \text{falls } w \notin L \end{cases}$ berechenbar ist.

Universelle Turing Maschine

eine Turing-Maschine, die jede Turing-Maschine simulieren kann, wenn deren Kodierung gegeben ist. Buchstaben des Bandalphabets als Wörter über $\{0,1,2\}$ mit $\Box = 2$ kodiert. Ab jetzt nehmen wir an, daß wir immer dieses Bandalphabet haben.
Eine Turing Maschine U heißt universelle Turing Maschine, wenn sie die folgende partielle Funktion berechnet. $\{0,1\}^* \rightarrow \{0,1\}^*$

$$y \rightarrow \begin{cases} \phi_w(x) & \text{falls } y = w000x, w \in L_{TM}, x \in \{0,1\}^* \\ undef. & \text{sonst} \end{cases}$$

Totale berechenbare Funktionen

Gesucht ist $C \subseteq TOT \subseteq L_{TM}$, so dass $\{\phi_w | w \in C\} = \{\phi_w | w \in TOT\}$ die Menge der totalen berechenbaren Funktionen ist.

- TOT ist nicht einmal semi-entscheidbar
- indirekt nehmen wir an, dass C semi-entscheidbar ist
- dann existiert eine totale berechenbare Funktion $f : \{0,1\}^* \rightarrow \{0,1\}^*$ mit Bild C
- neue Funktion $g : \{0,1\}^* \rightarrow \{0,1\}^* : w \vdash 1\phi_{f(w)}(w)$

Chomsky Normalform

Eine kontextfreie Grammatik G ist in Chomsky Normalform, falls alle Produktionen

- die Form $A \rightarrow AB$ oder $A \rightarrow a$ haben,
- und $S \rightarrow \epsilon$ und S nie auf der rechten Seite einer Produktion vorkommt