2.1.3 Wortproblem

Def (Wortproblem)

Gegeben eine Gram. $G = (V, \Sigma, P, S)$ und ein Wort $w \in \Sigma^*$, dann ist das Wortproblem zu entscheiden ob $w \in L(G)$.

Satz

Es existiert ein Algorithmus, der das Wortproblem für Typ 1 Spr. entscheidet.

Beweis: Sei $G = (V, \Sigma, P, S)$ eine Typ 1 Gram. Sei für $n, m \in \mathbb{N}$ die Menge T_m^n wie folgt def:

$$T_m^n = \{ w \in (V \cup \Sigma)^* \mid |w| \leq n \text{ und } w \text{ lässt sich aus } S \text{ in höchstens } m \text{ Schritten ableiten} \}$$

$$T_0^n = \{S\}$$

$$T_{m+1}^n = Abl_n(T_m^n)$$

$$Abl_G(X) = X \cup \{w \in (V \cup \Sigma)^* \mid |w| \leq n \text{ und } w' \Rightarrow_G w \text{ für ein Wort } w' \in X\}$$

Da nur endlich viele Worte mit $|w| \leq n$ gibt, ist auch $\bigcup_{m \geq 0} T_m^n$ für jedes n endlich. D.h. es gibt für jedes n ein m, so dass

$$T_m^n = T_{m+1}^n = T_{m+2}^n = \dots$$

Alg. 1. Für $w \in \Sigma^*$ mit $|w| = n$, berechne T_m^n für $m = T_{m+1}^n$

2. Teste $w \in T_m^n$.

$P = \{ S \to \varepsilon,\ S \to S',\ S' \to ab,\ S' \to aS'b \}$

$\Sigma = \{a, b\}$

$V = \{S, S'\}$

$aabb \in L$?

$T_0^4 = \{S\}$

$T_1^4 = \{S, \varepsilon, S'\}$

$T_2^4 = \{S, \varepsilon, S', ab, aS'b\}$

$T_3^4 = \{S, \varepsilon, S', ab, aS'b, aabb, \cancel{aaS'bb}\}$

$T_4^4 = T_3^4$

2.1.4 Syntaxbäume

Jeder Ableitung in einer Typ2-Grammatik kann man auch einen Syntaxbaum zuordnen. Für $x \in L(G)$ sei

$S = x_0 \Rightarrow x_1 \Rightarrow \dots \Rightarrow x_n = x$, die Ableitung von x.

Der Wurzel des Syntaxbaums wird S zugeordnet.

Für $i = 1, 2, \dots, n$: Falls in in der Schritt die Var A durch z ersetzt wird $(A \to z \in P)$ dann erhält der A entsprechende Knoten im Syntaxbaum $|z|$ Kinder, die mit den Zeichen aus z beschriftet werden.

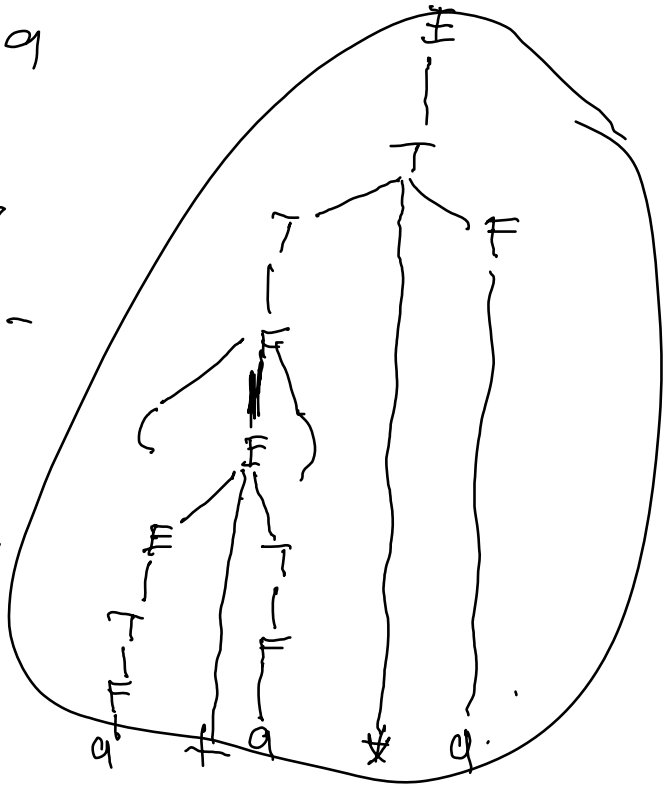
Bsp $G = (\{E, T, F\}, \{(,), a, +, *\}, P, E)$

$P = \{ E \to T,\ E \to E+T,\ T \to F,\ T \to T*F,\ F \to a,$
$F \to (E) \}$

$E \Rightarrow T \Rightarrow T*F \Rightarrow F*F \Rightarrow (E)*F \Rightarrow (E+T)*F \Rightarrow$
$(T+T)*F \Rightarrow (F+T)*F \Rightarrow (F+F)*F \Rightarrow (a+F)*F$
$(a+a)*F \Rightarrow (a+a)*a$

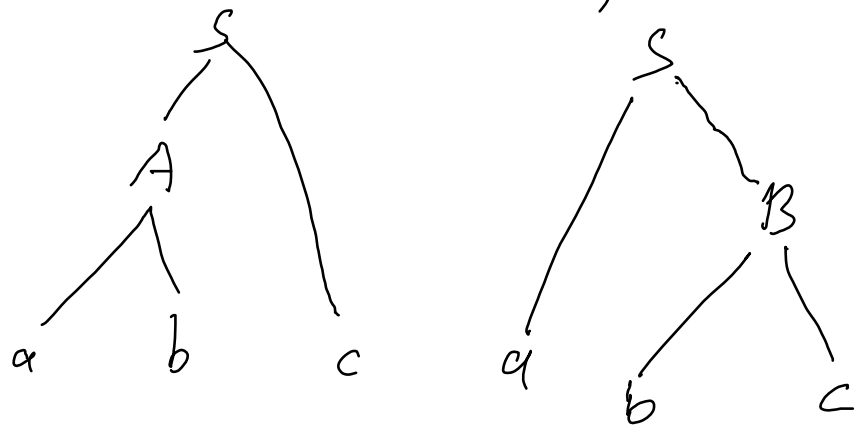
Bem: Verschiedene Ableitungen können zum gleichen Syntaxbaum führen.

Bem: In der Abl. haben wir immer das am weitesten links stehende Symbol ersetzt $\Rightarrow$ Linksableitung
Analog Rechtsableitung



Bem Zu einem Wort kann es versch. Syntax-
bäume geben.

Bsp $P = \{ S \to aB, \cancel{S \to Ac}, \cancel{A \to ab}, B \to bc \}$



Eine Gram G heißt mehrdeutig, wenn es für ein Wort $w \in L(G)$ mehrere verschiedene Syntaxbäume (= Linksableitungen) gibt.
Eine Spr. heißt inhärent mehrdeutig, wenn es keine Gram G mit $L(G) = A$ gibt, die nicht mehrdeutig ist.

Bsp.: $L = \{ a^i b^j c^k \mid i=j \text{ oder } j=k \}$

ist eine kontextfreie, inhärent mehrdeutige Spr.

$S \to I$

$S \to K$

$I \to AC$

$A \to aAb$

$A \to ab$

$C \to Cc$

$C \to c$

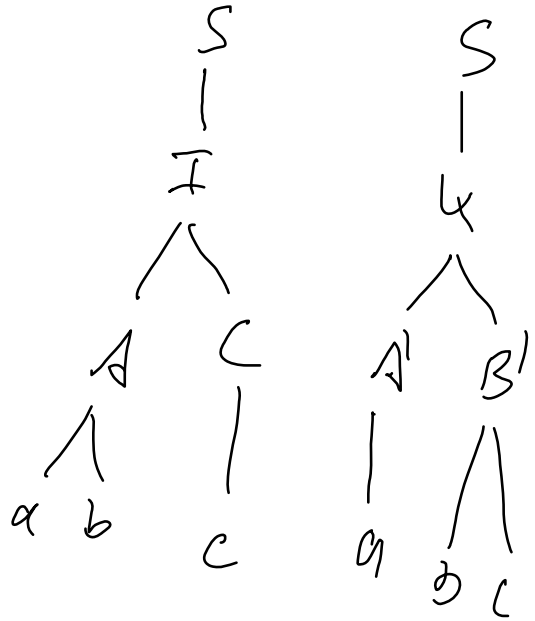
$K \to A'B'$

$A' \to a$

$A' \to A'a$

$B' \to bc$

$B \to bB'c$



2.1.5 BNF

Komplexe Beschreibungen von Prog.-Spr. werden durch BNF erreicht (erstmals ben. für ALGOL 60).

Falls:

$$A \to \alpha_1, \quad A \to \alpha_2, \ldots, A \to \alpha_n$$

dann schreibt man in BNF

$$A \to \alpha_1 \mid \alpha_2 \mid \ldots \mid \alpha_n$$

Oft auch "Erweiterte BNF" (EBNF)

$$A \to \alpha\gamma$$
$$A \to \alpha\beta\gamma$$

dann

$$A \to \alpha[\beta]\gamma$$

Falls

$A \to \alpha \gamma$

$A \to \alpha B \gamma$

$B \to \beta$

$B \to \beta B$

dann

$A \to \alpha \{ \beta \} \gamma$

In BNF statt "$\to$" "$::=$"

2.2 Reguläre Sprachen (Typ 3)

Typ 3- Spr $\hat{=}$ Sprachen, die von deterministischen endlichen Automaten akzeptiert werden

$\hat{=}$ Spr., die von nicht-det. endl. Aut. akzeptiert werden

$\hat{=}$ Spr., die durch reguläre Ausdrücke beschrieben werden,

Wichtige Eigenschaften:

- Pumping Lemma
- Abschlusseigenschaften

2.2.1 Endliche Automaten

Endliche Menge von Zuständen, endl. Eingabealphabet, endl. viele Zustandsübergänge.

Bsp: "Useless Machine"

Def (DEA)

Ein deterministischer endlicher Automat (DFA od. DEA) ist ein 5-Tupel:

$$M = (Z, \Sigma, \delta, z_0, E)$$

- Z: endl. Zustandsmenge
- Σ: Eingabealphabet mit $Z \cap \Sigma = \emptyset$
- $\delta: Z \times \Sigma \to Z$ Überführungsfkt.
- $z_0 \in Z$ Anfangszustand
- $E \subseteq Z$ Endzustände

Bsp $M = (Z, \Sigma, \delta, z_0, E)$ Zustandsgraph

$Z = \{z_0, z_1, z_2, z_3, z_4\}$

$\Sigma = \{a, h, !\}$

$\delta(z_0, h) = z_1$

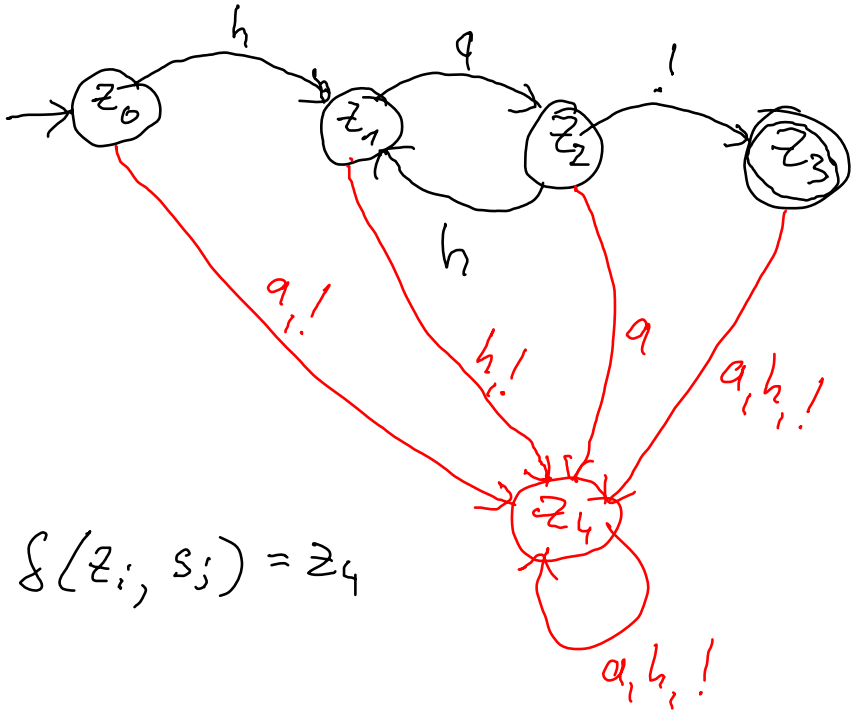
$\delta(z_1, a) = z_2$

$\delta(z_2, h) = z_1$

$\delta(z_2, !) = z_3$

für alle anderen z_i, s_j : $\delta(z_i, s_j) = z_4$

$E = \{z_3\}$



hahaha!h