TECHNISCHE UNIVERSITÄT DRESDEN

ALGORITHMEN UND DATENSTRUKTUREN

ÜBUNG 2: SYNTAXDIAGRAMME & EBNF

Eric Kunze

eric.kunze@tu-dresden.de

TU Dresden, 06.11.2020

VIDEOEMPFEHLUNG

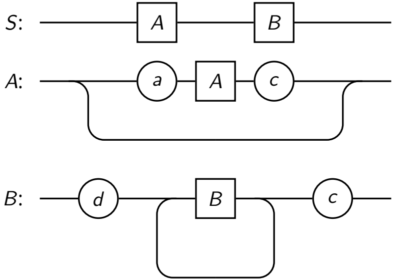
Prof. Dr. Markus Krötzsch hat im vergangenen Wintersemester 2020/21 die Vorlesung „Formale Systeme“ (3. Semester) in Form von YouTube-Videos gehalten. Diese Vorlesung beschäftigt sich vertieft mit formalen Sprachen.

Die Einleitung entspricht ungefähr dem Inhalt der ersten Übung:

- `https://youtu.be/Lma6jaPnD-I`

Syntaxdiagramme

Beispiel eines Syntaxdiagrammsystems mit Startdiagramm *S*:



A . . . Nichtterminalsymbol = syntaktische Variable

a . . . Terminalsymbol

RÜCKSPRUNGALGORITHMUS

Rücksprungalgorithmus

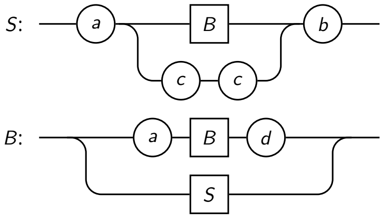
- Ziel: Nachweis von Zugehörigkeit eines Wortes zu einer Sprache
- jedes Kästchen bekommt eindeutige Marke (Rücksprungadresse)
- beim Betreten eines Syntaxdiagramms wird eine Marke auf den Keller gelegt

Hauptaugenmerk:
Protokollierung von Wortentstehung & Markenkeller

- jede Zeile entspricht dem Aufenthalt in einem Syntaxdiagramm
- jede Zeile führt eine Operation auf dem Markenkeller durch

AUFGABE 1

Gegeben sei das folgende Syntaxdiagrammsystem $\mathcal{U}$ mit Startdiagramm S:



Beispiele für Wörter, die das System $\mathcal{U}$ erzeugt:

- a $accb$ b
- a a $accb$ b b
- a a $accb$ d b
- a a a $accb$ d d b
- a a a $accb$ b d b

AUFGABE 1 — TEIL (B)

Wort: `aaaaccbdbb`

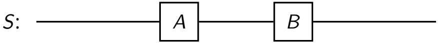
Protokollierungszeitpunkte:

- jeder Aufenthalt in einem Syntaxdiagramm entspricht einer Zeile
- jede Zeile führt eine Operation auf dem Markenkeller aus
- $\cancel{3}$ = Rücksprung zu Marke 3

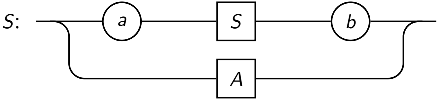
Wort	Markenkeller
a	1
a	31
aa	131
aaa	2131
aaa	32131
aaaaccb	$\cancel{3}2131$
aaaaccb	$\cancel{2}131$
aaaaccbd	$\cancel{1}31$
aaaaccbdb	$\cancel{3}1$
aaaaccbdb	$\cancel{1}$
aaaaccbdbb	–

GRUNDKONSTRUKTION VON SYNTAXDIAGRAMMEN

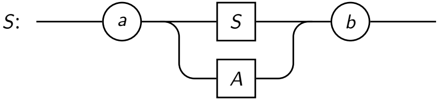
$L = L_A \cdot L_B$



$L = \{a^n \; L_A \; b^n : n \geq 0\}$



$L = \{a^n \; L_A \; b^n : n > 0\}$

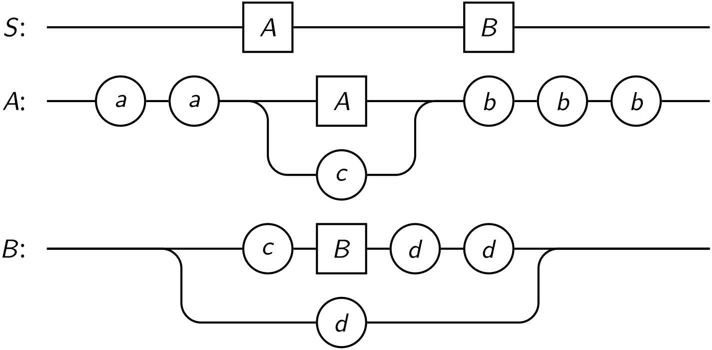


kleine Tricks:

- $a^{2n} = (a^2)^n = (aa)^n$
- $a^{2n+1} = a \; a^{2n} = a \; (aa)^n$

AUFGABE 1 — TEIL (C)

$$
\begin{aligned}
L &= \left\{ a^{2i} c b^{3i} c^k d^{2k+1} \mid i > 0, k \geq 0 \right\} \\
&= \left\{ a^{2i} c b^{3i} \mid i > 0 \right\} \cdot \left\{ c^k d^{2k+1} \mid k \geq 0 \right\} \\
&= \left\{ (aa)^i c (bbb)^i \mid i > 0 \right\} \cdot \left\{ c^k d (dd)^k \mid k \geq 0 \right\}
\end{aligned}
$$



Extended Backus-Naur-Form

EBNF-DEFINITION

- EBNF-Definition besteht aus endlicher Menge von EBNF-Regeln.

$$\mathcal{E} = (V, \Sigma, S, R)$$

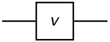
- Jede EBNF-Regel besteht aus einer linken und einer rechten Seite, die rechte Seite ist ein EBNF-Term.

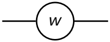
$$\mathit{left} ::= \mathit{EBNFTerm}$$

Definition (EBNF-Terme): Seien V (syntaktische Variablen) und Σ (Terminalsymbole) endliche Mengen mit $V \cap \Sigma = \emptyset$. Die Menge der EBNF-Terme über V und Σ (notiere: $T(\Sigma, V)$), ist die *kleinste* Menge $T \subseteq \left(V \cup \Sigma \cup \left\{ \hat{\{}, \hat{\}}, \hat{[}, \hat{]}, \hat{(}, \hat{)}, \hat{|} \right\} \right)$ mit $V \subseteq T$, $\Sigma \subseteq T$ und

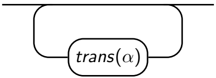
- Wenn $\alpha \in T$, so auch $\hat{(}\ \alpha\ \hat{)} \in T$, $\hat{\{}\ \alpha\ \hat{\}} \in T$, $\hat{[}\ \alpha\ \hat{]} \in T$.
- Wenn $\alpha_1, \alpha_2 \in T$, so auch $\hat{(}\ \alpha_1\ \hat{|}\ \alpha_2\ \hat{)} \in T$, $\alpha_1 \alpha_2 \in T$.

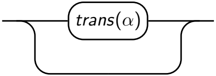
ÜBERSETZUNG EBNF $\leftrightarrow$ SYNTAXDIAGRAMME

Sei $v \in V$ und $w \in \Sigma$. $\mathit{trans}(v) =$ 

; $\mathit{trans}(w) =$ 

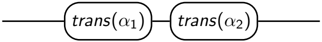
Sei $\alpha \in T(\Sigma, V)$ ein EBNF-Term.

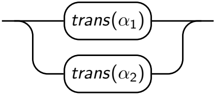
- $\mathit{trans}(\ \hat{\{}\ \alpha\ \hat{\}}\ ) =$ 

- $\mathit{trans}(\ \hat{[}\ \alpha\ \hat{]}\ ) =$ 

- $\mathit{trans}(\ \hat{(}\ \alpha\ \hat{)}\ ) = \mathit{trans}(\alpha)$

Seien $\alpha_1, \alpha_2 \in T(\Sigma, V)$ zwei EBNF-Terme.

- $\mathit{trans}(\ \alpha_1\alpha_2\ ) =$ 

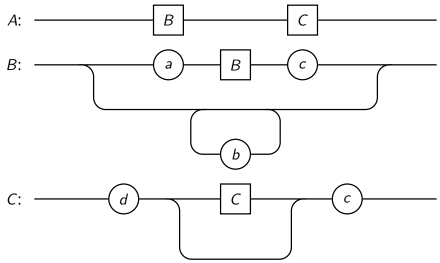
- $\mathit{trans}(\ \hat{(}\ \alpha_1\ \hat{|}\ \alpha_2\ \hat{)}\ ) =$ 

AUFGABE 2 — TEIL (A)

EBNF-Definition $\mathcal{E} = (V, \Sigma, A, R)$ mit $\Sigma = \{a, b, c, d\}$,

$$V = \{A, B, C\} \quad \text{und} \quad R = \Big\{ A ::= BC,\quad B ::= \hat{(}\; aBc \;\hat{|}\; \hat{\{}\; b \;\hat{\}}\; \hat{)},\quad C ::= d \;\hat{[}\; C \;\hat{]}\; c \Big\}$$

Übersetzung in ein Syntaxdiagrammsystem:



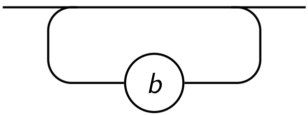
Startdiagramm: A

AUFGABE 2 — TEIL (B)

Wir wollen die von $\mathcal{E}$ beschriebene Sprache L_A beschreiben und wenden dafür die Grundkonstruktionen „rückwärts“ an.

$$
\begin{aligned}
L_A &= L_B \cdot L_C \\
&= \{a^n\ w\ c^n : w \in \{b\}^* : n \geq 0\} \cdot \{d^m c^m : m \geq 1\}
\end{aligned}
$$

Der Teil $w \in \{b\}^*$ beschreibt dabei, dass wir ein beliebiges Wort aus $\{b\}^*$ schreiben. Diese Sprache $\{b\}^*$ wird durch



beschrieben.

AUFGABE 2 — TEIL (C)

Gegeben sei die Sprache

$$L = \left\{ a^{n+\ell} c b^n (cd)^\ell : n, \ell \in \mathbb{N}, n \geq 1 \right\}$$

Gesucht ist eine zugehörige EBNF-Definition.

$$L = \left\{ a^\ell a^n c b^n (cd)^\ell : n, \ell \in \mathbb{N}, n \geq 1 \right\}$$

Lösungsweg: via Syntaxdiagrammsystem & Übersetzung

EBNF-Definition: $\mathcal{E}' = (V, \Sigma, S, R)$ mit $\Sigma = \{a, b, c, d\}$,

$$V = \{S, A\} \quad \text{und} \quad R = \left\{ \begin{array}{l} S ::= \hat{(}\ aScd\ \hat{|}\ A\ \hat{)}, \\ A ::= a\ \hat{(}\ A\ \hat{|}\ c\ \hat{)}\ b \end{array} \right\}$$