... haben Regeln der Form $A \longrightarrow \alpha, A \subseteq V, \alpha \in (V \cup \sum)^*$

Definition 1 *Eine kontextfreie Grammatik* (Σ, V, P, S) *ist in Chomsky Normalform, wenn jede Regel in* P *von der Form*

$$\begin{array}{rcccl} & v & \longrightarrow & uz & u, z \in V \backslash \{S\} \\ \textit{oder} & v & \longrightarrow & a & a \in \Sigma \end{array}$$

ist. Zusätzlich ist die Regel $S \rightarrow \epsilon$ *erlaubt.*

Es gilt folgender Satz.

Satz 2 *Sei* G *eine kontextfreie Grammatik. Dann kann aus* G *eine kontextfreie Grammatik* G' *in Chomsky Normalform konstruiert werden, die dieselbe Sprache wie* G *erzeugt.*

HEINZ NIXDORF INSTITUT
Universität Paderborn
Algorithmen und Komplexität

… ist entscheidbar.

Sei G kontextfreie Grammatik für L in Chomsky Normal Form.
Dann hat jede Ableitung für ein Wort x aus L, $|x| = n$
die Länge $2n - 1$.
Es gibt nur endlich viele Ableitungen der Länge $2n-1$.

$\longrightarrow$ **Naiver Algorithmus**:
Probiere alle diese Ableitungen aus.

Zeit: Exponentiell in n.

HEINZ NIXDORF INSTITUT
Universität Paderborn
Algorithmen und Komplexität

Der CYK-Algorithmus.
Eingabe: Kontextfreie Grammatik G in Chomsky-NF, $w \in \Sigma^*$.

Frage: $w \in L(G)$?

CYK-Algorithmus: Eingabe $w_1\, w_2 \ldots w_n \in \Sigma^*$

Berechne für jedes Paar i, j, $1 \le i \le j \le n$, die Menge
$T(i,j) = \{A \in V, A \to w_i \ldots w_j\}$,
und zwar zuerst für alle i, j mit
$j - i = 0$, dann $j - i = 1$, dann $j - i = 2$, …, $j - i = n - 1$.

Für $j - i = n - 1$ ist $i = 1$, $j = n$.
Also: $w \in L \iff S \in T(1, n)$.

CYK-Algorithmus/Fortsetzung

Wie berechnet man T (i,j)?

$T(i,i) = \{A, A \vdash^{*} w_i\} = \{A, A \longrightarrow w_i\} \longrightarrow$ In P nachlesen.

Seien alle T (i,j) mit j - i < r berechnet.

Wie berechnet man T (i,j) mit j – i = r?

Falls $A \in T(i,j)$ ist, gibt es

$A \longrightarrow BC \vdash^{*} w_i \dots w_k \; w_{k+1} w_j$ mit

$B \vdash^{*} w_i \dots w_k$, $k\text{-}i < r$, und $C \vdash^{*} w_{k+1} \dots w_j$ $j - k+1 < r$

d.h.: Es gibt k, $i \leq k < j$ und $(A \longrightarrow BC) \in P$

mit: $B \in T(i,k)$, $C \in T(k+1, j)$,

Beachte: T(i,k), T (k+1, j) sind bereits berechnet!

CYK-Algorithmus/Fortsetzung

Laufzeit: Für T (i,j): $O((j - i + 1) \cdot |P| \cdot |V|)$

Insgesamt: $O(\sum_{1 \leq i \leq j \leq n} (j - i + 1) |P| \cdot |V|) = O(n^3 \cdot |P| \cdot |V|)$

Polynomielle Laufzeit, aber viel zu langsam, um ihn etwa für die Syntaxanalyse bei (Java-)Programmen einzusetzen.

$\longrightarrow$

Hier arbeitet man mit eingeschränkten Teilklassen von CF, den **deterministisch kontextfreien Sprachen**, gut strukturierten Grammatiken und zugehörigen Parsern.

Sei L kontextfrei. Dann gibt es ein $p \in \mathbb{N}$, so dass für alle $w \in L$ mit $|w| \geq p$ eine Zerlegung $w = uvxyz$ existiert mit:

(i) $|v\,x\,y| \leq p$

(ii) $|vy| \geq 1$

(iii) $u\,v^i\,x\,y^i\,z \in L$ für alle $i \geq 0$.

Anwendungen des Pumping Lemma

Wir wenden die Kontraposition des Pumping Lemma an:

Falls für jedes $p \in \mathbb{N}$ ein $w \in L$ mit $|w| \geq p$ existiert, so dass für jede Zerlegung $w = uvxyz$ mit $|vy| \geq 1$ und $|vxy| \leq p$ ein i existiert mit $u\,v^i\,x\,y^i\,z \notin L$,

so ist L nicht kontextfrei.

Folgende Sprachen sind nicht kontextfrei:

$L_1 = \{a^n\, b^n\, c^n \mid n \geq 1\}$

$L_2\ \{a^i\, b^j\, c^k \mid 1 \leq i \leq j \leq k\}$

Korollar: Sei CF die Klasse der kontextfreien, CS die Klasse der kontextsensitiven Sprachen.
Dann ist CF eine echte Teilmenge von CS.

Bew.: „$\subseteq$“ ist klar. „$\neq$„ gilt, da $L_1 \in$ CS, aber $L_1 \notin$ CF gilt.

Abschlusseigenschaften kontextfreier Sprachen

Seien L und L‘ kontextfrei. Dann sind auch

(i) $L \cup L'$

(ii) L L‘

(iii) L^*

kontextfrei.

Aber: Es gibt kontextfreie L, L’, so dass $L \cap L'$ nicht kontextfrei ist.

Bsp: $L = \{a^i\, b^j\, c^j,\ i, j \geq 1\}$, $L' = \{a^i\, b^i\, c^j,\ i, j \geq 1\}$

Es gibt kontextfreie L, so dass $\bar{L}$ nicht kontextfrei ist.