Theoretische Grundlagen der Informatik
Vorlesung am 8. Januar 2019

INSTITUT FÜR THEORETISCHE INFORMATIK

Letzte Vorlesung

Eine **Grammatik** ist $G = (\Sigma, V, S, R)$ mit

- Alphabet Σ aller Terminale,
- Menge V aller Variablen / Nichtterminale,
- Startvariable S aus V,
- Menge R aller Regeln / Ableitungen, wobei eine Ableitung ein Tupel (ℓ, r) aus Wörtern in $(\Sigma \cup V)^*$ ist.

Wir schreiben auch $\ell \to r$ für die Regel (ℓ, r) und entsprechend $S \xrightarrow{*} w$ falls w aus S abgeleitet werden kann.

Die **erzeugte Sprache** ist $L(G) = \{w \in \Sigma^* \mid S \xrightarrow{*} w\}$.

Eine Grammatik ist **kontextfrei** oder **Chomsky Typ-2**, wenn alle Regeln die folgende Form haben:

$$A \to v \quad \text{mit } A \in V \text{ und } v \in (\Sigma \cup V)^*$$

Letzte Vorlesung

Beispiel

Grammatik $G = (\Sigma, V, S, R)$ mit

$$\begin{array}{rcl} V & = & \{S, A, B\} \\ \Sigma & = & \{0, 1\} \\ R & = & \{S \to 0B \mid 1A, \\ & & \quad A \to 0 \mid 0S \mid 1AA, \\ & & \quad B \to 1 \mid 1S \mid 0BB\} \end{array}$$

erzeugt die Sprache $L = L(G) \subseteq \{0, 1\}^+$ aller Wörter mit genauso vielen Einsen wie Nullen.

Zum Beispiel ist $110010 \in L(G)$, wegen der Ableitung

$$S \to 1A \to 11AA \to 11A0 \to 110S0 \to 1100B0 \to 110010.$$

Heute

- Syntaxbäume
- Eindeutigkeit
- Chomsky-Normalform
- CYK-Algorithmus

Syntaxbäume

Eine Grammatik ist **kontextfrei** oder **Chomsky Typ-2**, wenn alle Regeln die folgende Form haben:

$$A \to v \quad \text{mit } A \in V \text{ und } v \in (\Sigma \cup V)^*$$

Syntaxbäume visualisieren die Ableitung eines einzelnen Wortes in einer kontextfreien Grammatik.

- An der Wurzel eines Syntaxbaumes steht das Startsymbol.
- Jeder innere Knoten enthält eine Variable.
- Die Blätter sind Symbole aus Σ oder ε.
- Wenn ein innerer Knoten A als Nachfolger von links nach rechts $\alpha_1, \ldots, \alpha_r \in V \cup \Sigma$ hat, so muss $A \to \alpha_1 \ldots \alpha_r$ eine Ableitungsregel der Grammatik sein.

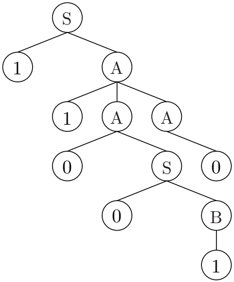
Syntaxbäume – Beispiel

Zu den Regeln

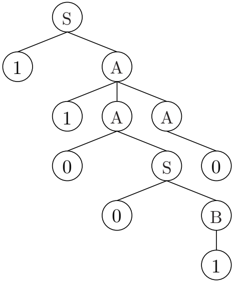
$$R = \{S \to 0B \mid 1A, \quad A \to 0 \mid 0S \mid 1AA, \quad B \to 1 \mid 1S \mid 0BB\}$$

betrachte die Ableitung

$$S \to 1A \to 11AA \to 11A0 \to 110S0 \to 1100B0 \to 110010.$$



Syntaxbäume – Beispiel



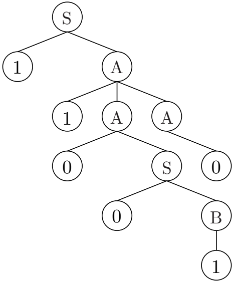
- Zu jeder Ableitung gehört genau ein Syntaxbaum.
- Zu einem Syntaxbaum können jedoch verschiedene Ableitungen des gleichen Wortes gehören.

Syntaxbäume – Beispiel

$$R = \{S \to 0B \mid 1A,\ A \to 0 \mid 0S \mid 1AA,\ B \to 1 \mid 1S \mid 0BB\}$$

$$S \to 1A \to 11AA \to 110SA \to 1100BA \to 11001A \to 110010$$

$$S \to 1A \to 11AA \to 11A0 \to 110S0 \to 1100B0 \to 110010$$



Links/Rechtsableitung, Eindeutigkeit

Für Chomsky-2 Grammatiken gilt:

- Wegen der Kontextfreiheit ist die Reihenfolge, in der abgeleitet wird, für das Ergebnis unerheblich.

Eine **Linksableitung** (**Rechtsableitung**) ist eine Ableitung, bei der in jedem Schritt die linkeste (rechteste) Variable abgeleitet wird.

Eine kontextfreie Grammatik G heißt **eindeutig**, wenn es für jedes Wort $w \in L(G)$ genau einen Syntaxbaum gibt.

Eine kontextfreie Sprache L heißt **eindeutig**, wenn es eine eindeutige Grammatik G mit $L(G) = L$ gibt. Ansonsten heißt L **inhärent mehrdeutig**.

Beispiel

Karlsruher Institut für Technologie

Die Sprache

$$L = \{0^n 1^n \mid n \geq 1\}$$

erzeugt durch die Grammatik

$$\begin{aligned} V &= \{S\} \\ \Sigma &= \{0, 1\} \\ R &= \{S \to 01 \mid 0S1\} \end{aligned}$$

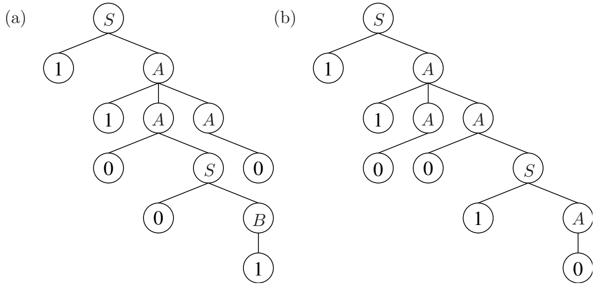
ist eindeutig.

Beispiel

Die Sprache gegeben durch die Regeln

$$R = \{S \to 0B \mid 1A, A \to 0 \mid 0S \mid 1AA, B \to 1 \mid 1S \mid 0BB\}$$

ist nicht eindeutig.



Heute

- Syntaxbäume
- Eindeutigkeit
- Chomsky-Normalform
- CYK-Algorithmus

Chomsky-Normalform

Eine Grammatik ist **kontextfrei** oder **Chomsky Typ-2**, wenn alle Regeln die folgende Form haben:

$$A \to v \quad \text{mit } A \in V \text{ und } v \in (\Sigma \cup V)^*$$

Eine kontextfreie Grammatik ist in **Chomsky-Normalform**, wenn alle Regeln von der Form:

$$A \to BC \qquad \text{oder} \qquad A \to a$$

sind, mit $A, B, C \in V$ und $a \in \Sigma$.

Chomsky-Normalform

Eine kontextfreie Grammatik ist in **Chomsky-Normalform**, wenn alle Regeln von der Form:

$$A \to BC \qquad \text{oder} \qquad A \to a$$

sind, mit $A, B, C \in V$ und $a \in \Sigma$.

- Grammatiken in Chomsky-Normalform können also nicht das Wort ε erzeugen.
- Für kontextfreie Sprachen, die ε enthalten, lässt sich eine Grammatik in Chomsky-Normalform leicht erweitern durch die Regeln

  $$S' \to \varepsilon \qquad \text{und} \qquad S' \to S,$$

  wobei S' ein neues Startsymbol zur Erzeugung von ε ist.

Chomsky-Normalform

Satz:
Jede kontextfreie Grammatik kann in eine Grammatik in erweiterter Chomsky-Normalform überführt werden.

Eine Grammatik in erweiterter Chomsky-Normalform ist eine Grammatik in Chomsky-Normalform mit den zusätzlichen Regeln $S' \to \varepsilon$ und $S' \to S$.

Beweis (konstruktiv):

- Wir geben eine Schritt–für–Schritt–Überführung der Regeln einer beliebigen kontextfreien Grammatik in Regeln in Normalform an.
- Großbuchstaben repräsentieren immer Variablen / Nichtterminale.
- Kleinbuchstaben repräsentieren immer Terminale.

Chomsky-Normalform

Wir veranschaulichen den Beweis an folgendem Beispiel:
Grammatik $G = (\Sigma, V, S, R)$ mit

$$\begin{aligned} \Sigma &= \{a, b\} \\ V &= \{A, B, C, D, E, S\} \end{aligned}$$

und der folgenden Regelmenge R:

$$\begin{aligned} S &\rightarrow A \mid aAa \mid bBb \mid \varepsilon \\ A &\rightarrow C \mid a \\ B &\rightarrow b \\ C &\rightarrow AF \mid CDE \mid \varepsilon \\ D &\rightarrow A \mid B \mid ab \\ E &\rightarrow B \\ F &\rightarrow D \mid E \end{aligned}$$

Schritt 1

Ziel:

- Alle Regeln enthalten auf der rechten Seite nur Symbole aus V oder nur ein Symbol aus Σ.

Vorgehen:

- Ersetze dazu in allen rechten Seiten von Regeln Symbole $a \in \Sigma$ durch neue Variablen Y_a und füge die Regeln $Y_a \to a$ hinzu.

Schritt 1

Ziel:

- Alle Regeln enthalten auf der rechten Seite nur Symbole aus V oder nur ein Symbol aus Σ.

Vorgehen:

- Ersetze dazu in allen rechten Seiten von Regeln Symbole $a \in \Sigma$ durch neue Variablen Y_a und füge die Regeln $Y_a \to a$ hinzu.

$$
\begin{array}{lcl}
S & \to & A \mid aAa \mid bBb \mid \varepsilon \\
A & \to & C \mid a \\
B & \to & b \\
C & \to & AF \mid CDE \mid \varepsilon \\
D & \to & A \mid B \mid ab \\
E & \to & B \\
F & \to & D \mid E
\end{array}
\quad \Rightarrow \quad
\begin{array}{lcl}
S & \to & A \mid Y_aAY_a \mid Y_bBY_b \mid \varepsilon \\
A & \to & C \mid a \\
B & \to & b \\
C & \to & AF \mid CDE \mid \varepsilon \\
D & \to & A \mid B \mid Y_aY_b \\
E & \to & B \\
F & \to & D \mid E \\
Y_a & \to & a \\
Y_b & \to & b
\end{array}
$$

Schritt 2

Ziel:

- Alle rechten Seiten haben Länge ≤ 2.

Vorgehen:

- Sei $A \to B_1 \dots B_m$ Regel mit $m > 2$.
- Führe $m-2$ neue Variablen $C_1, \dots, C_{m-2}$ ein, und ersetze die Regel

$$A \to B_1 \dots B_m$$

durch neue Regeln

$$\begin{aligned} A &\to B_1 C_1 \\ C_i &\to B_{i+1} C_{i+1} \qquad \text{für } 1 \leq i \leq m-3 \\ C_{m-2} &\to B_{m-1} B_m \end{aligned}$$

Schritt 2

Karlsruher Institut für Technologie

Ziel:

- Alle rechten Seiten haben Länge ≤ 2.

$$
\begin{array}{lcl}
S & \rightarrow & A \mid Y_aAY_a \mid Y_bBY_b \mid \varepsilon \\
A & \rightarrow & C \mid a \\
B & \rightarrow & b \\
C & \rightarrow & AF \mid CDE \mid \varepsilon \\
D & \rightarrow & A \mid B \mid Y_aY_b \\
E & \rightarrow & B \\
F & \rightarrow & D \mid E \\
 & & \\
 & & \\
 & & \\
Y_a & \rightarrow & a \\
Y_b & \rightarrow & b
\end{array}
\quad \Rightarrow \quad
\begin{array}{lcl}
S & \rightarrow & A \mid Y_aC_1 \mid Y_bC_2 \mid \varepsilon \\
A & \rightarrow & C \mid a \\
B & \rightarrow & b \\
C & \rightarrow & AF \mid CC_3 \mid \varepsilon \\
D & \rightarrow & A \mid B \mid Y_aY_b \\
E & \rightarrow & B \\
F & \rightarrow & D \mid E \\
C_1 & \rightarrow & AY_a \\
C_2 & \rightarrow & BY_b \\
C_3 & \rightarrow & DE \\
Y_a & \rightarrow & a \\
Y_b & \rightarrow & b
\end{array}
$$

Schritt 3

Ziel:

- Es kommen keine Regeln $A \to \varepsilon$ vor.

Vorgehen, Phase 1:
Finde die Menge V' aller Variablen A für die $A \xrightarrow{*} \varepsilon$ existiert:

- Es werden erst alle A mit $A \to \varepsilon$ in V' aufgenommen.
- Dann wird geprüft, ob neue Regeln $B \to \varepsilon$ entstehen, wenn man A in allen Regeln auf der rechten Seite A durch ε ersetzt.
- Ist dies der Fall, so werden die entsprechenden Variablen B in V' aufgenommen und genauso behandelt.
- Das Verfahren hört auf, wenn V' sich nicht mehr ändert.

Schritt 3

Ziel:

- Es kommen keine Regeln $A \to \varepsilon$ vor.

Vorgehen, Phase 1:
Finde die Menge V' aller Variablen A für die $A \xrightarrow{*} \varepsilon$ existiert:

- Bemerkung: In Phase 1 werden noch keine Regeln geändert.
- Die Ersetzung ist also nur “testweise”.
- Am Ende enthält V' alle Variablen A mit $A \xrightarrow{*} \varepsilon$.

Schritt 3

Ziel:

- Es kommen keine Regeln $A \to \varepsilon$ vor.

Vorgehen, Phase 2: Ersetzung.

- Gegeben V' aus Phase 1.
- Streiche alle Regeln $A \to \varepsilon$.
- Für $A \to BC$ füge die zusätzliche Regel
  - $A \to B$ falls $C \in V'$,
  - $A \to C$ falls $B \in V'$

  ein.
- (Die Regel $A \to BC$ wird nicht gestrichen).

Schritt 3

- Initialisierung
  $V' = \{A \mid A \to \varepsilon\} = \{S, C\}$
- Erster Durchlauf liefert: A
  $\Rightarrow V' = \{A, C, S\}$
- Zweiter Durchlauf liefert: D
  $\Rightarrow V' = \{A, C, D, S\}$
- Dritter Durchlauf liefert: F
  $\Rightarrow V' = \{A, C, D, F, S\}$
- Vierter Durchlauf liefert nichts neues.

$$
\begin{array}{rcl}
S & \to & A \mid Y_a C_1 \mid Y_b C_2 \mid \varepsilon \\
A & \to & C \mid a \\
B & \to & b \\
C & \to & AF \mid CC_3 \mid \varepsilon \\
D & \to & A \mid B \mid Y_a Y_b \\
E & \to & B \\
F & \to & D \mid E \\
C_1 & \to & AY_a \\
C_2 & \to & BY_b \\
C_3 & \to & DE \\
Y_a & \to & a \\
Y_b & \to & b
\end{array}
$$

Schritt 3

- Streiche ε-Produktionen
- Simuliere Ableitungen $A \xrightarrow{*} \varepsilon$ auf den verbleibenden Regeln $V' = \{A, C, D, F, S\}$

$$
\begin{aligned}
S &\rightarrow A \mid Y_a C_1 \mid Y_b C_2 \\
A &\rightarrow C \mid a \\
B &\rightarrow b \\
C &\rightarrow AF \mid F \mid A \mid CC_3 \mid C_3 \\
D &\rightarrow A \mid B \mid Y_a Y_b \\
E &\rightarrow B \\
F &\rightarrow D \mid E \\
C_1 &\rightarrow AY_a \mid Y_a \\
C_2 &\rightarrow BY_b \\
C_3 &\rightarrow DE \mid E \\
Y_a &\rightarrow a \\
Y_b &\rightarrow b
\end{aligned}
$$

Schritt 4

Ziel

- Die Grammatik enthält keine (Ketten-)Regeln der Form $A \to B$.
- Beispiel

$$\begin{aligned} A &\to B \mid C \\ B &\to C \\ C &\to c \end{aligned}$$

Schritt 4

Ziel

- Die Grammatik enthält keine (Ketten-)Regeln der Form $A \to B$.

Vorgehen

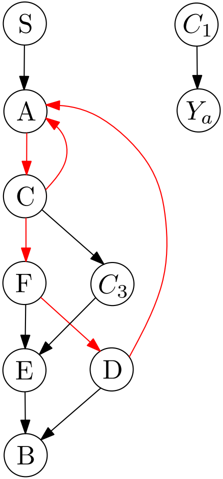
- **Phase 1:** Finde alle Kreise $A_1 \to A_2 \to \dots \to A_n \to A_1$.
  Ersetze alle A_i durch A_1.
- **Phase 2:** Betrachte die Regeln der Form $A \to B$ in umgekehrt topologischer Reihenfolge.
  - Für jede Regel $A \to B$ und jede Regel $B \to \beta$ füge Regel $A \to \beta$ hinzu.
  - Lösche Regel $A \to B$.

Schritt 4

Topologische Sortierung der Regelmenge

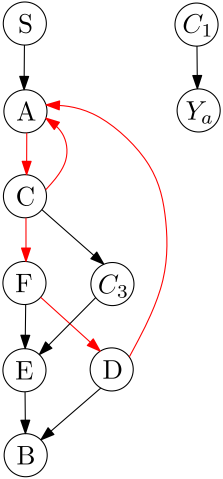
- $V_1, \ldots, V_k$ Menge von Variablen aus Kettenregeln
- $V_1, \ldots, V_k$ topologisch sortiert, wenn gilt:
- $V_i \xrightarrow{*} V_j \Rightarrow i < j$
- Voraussetzung: Es gibt keine zyklischen Abhängigkeiten

Abhängigkeitsgraph



$$
\begin{aligned}
S &\rightarrow A \mid Y_aC_1 \mid Y_bC_2 \\
A &\rightarrow C \mid a \\
B &\rightarrow b \\
C &\rightarrow AF \mid F \mid A \mid CC_3 \mid C_3 \\
D &\rightarrow A \mid B \mid Y_aY_b \\
E &\rightarrow B \\
F &\rightarrow D \mid E \\
C_1 &\rightarrow AY_a \mid Y_a \\
C_2 &\rightarrow BY_b \\
C_3 &\rightarrow DE \mid E \\
Y_a &\rightarrow a \\
Y_b &\rightarrow b
\end{aligned}
$$

Schritt 4 – Phase 1



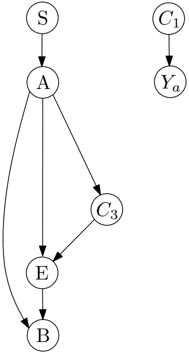
- Zyklen $A \to C$ und $A \to C \to F \to D \to A$
- $\Rightarrow A, C, F, D$ äquivalent
- Entferne an Zyklen beteiligte Regeln
- Ersetze Vorkommen von C, F, D in allen Regeln durch A
- Lösche Regeln der Form $A \to A$

Schritt 4 – Phase 1

Zyklus: $A \to C \to F \to D \to A$

$$
\begin{array}{rcl}
S & \to & A \mid Y_a C_1 \mid Y_b C_2 \\
A & \to & C \mid a \\
B & \to & b \\
C & \to & AF \mid F \mid A \mid CC_3 \mid C_3 \\
D & \to & A \mid B \mid Y_a Y_b \\
E & \to & B \\
F & \to & D \mid E \\
C_1 & \to & AY_a \mid Y_a \\
C_2 & \to & BY_b \\
C_3 & \to & DE \mid E \\
Y_a & \to & a \\
Y_b & \to & b
\end{array}
\quad \Rightarrow \quad
\begin{array}{rcl}
S & \to & A \mid Y_a C_1 \mid Y_b C_2 \\
A & \to & a \\
B & \to & b \\
A & \to & AA \mid AC_3 \mid C_3 \\
A & \to & B \mid Y_a Y_b \\
E & \to & B \\
A & \to & E \\
C_1 & \to & AY_a \mid Y_a \\
C_2 & \to & BY_b \\
C_3 & \to & AE \mid E \\
Y_a & \to & a \\
Y_b & \to & b
\end{array}
$$

Schritt 4 – Phase 2



$$
\begin{aligned}
S &\to A \mid Y_a C_1 \mid Y_b C_2 \\
A &\to a \mid AA \mid AC_3 \mid C_3 \mid B \mid Y_a Y_b \mid E \\
B &\to b \\
E &\to B \\
C_1 &\to AY_a \mid Y_a \\
C_2 &\to BY_b \\
C_3 &\to AE \mid E \\
Y_a &\to a \\
Y_b &\to b
\end{aligned}
$$

Topologische Sortierung: $S, A, C_3, E, B, C_1, Y_a$

Schritt 4 – Phase 2

- Gehe in umgekehrter topologischer Sortierung vor.
- Ersetze $A \to B$ durch $A \to \beta$, falls Regel $B \to \beta$ existiert.

Topologische Sortierung: $S, A, C_3, E, B, C_1, Y_a$

$$\begin{aligned}
S &\to A \mid Y_aC_1 \mid Y_bC_2 \\
A &\to a \mid AA \mid AC_3 \mid C_3 \mid \\
&\quad B \mid Y_aY_b \mid E \\
B &\to b \\
E &\to B \\
C_1 &\to AY_a \mid Y_a \\
C_2 &\to BY_b \\
C_3 &\to AE \mid E \\
Y_a &\to a \\
Y_b &\to b
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
S &\to Y_aC_1 \mid Y_bC_2 \mid a \mid AA \\
&\quad AC_3 \mid AE \mid b \mid Y_aY_b \\
A &\to a \mid AA \mid AC_3 \mid AE \mid \\
&\quad b \mid Y_aY_b \\
B &\to b \\
E &\to b \\
C_1 &\to AY_a \mid a \\
C_2 &\to BY_b \\
C_3 &\to AE \mid b \\
Y_a &\to a \\
Y_b &\to b
\end{aligned}$$

Schritt 4 – Phase 2

- Gehe in umgekehrter topologischer Sortierung vor.
- Ersetze $A \to B$ durch $A \to \beta$, falls Regel $B \to \beta$ existiert.

Testen Sie sich:
Warum gehen wir in dieser Reihenfolge vor?

Sonderbehandlung von ε-Produktionen

- Grammatik ist nun in Chomsky-Normalform, aber
- G enthält Ableitungsfolge $S \overset{*}{\rightarrow} \varepsilon$ (haben wir entfernt).
- Erweitere Grammatik um Regeln $S' \rightarrow S$ und $S' \rightarrow \varepsilon$ für neues Startsymbol S'.

Heute

- Syntaxbäume
- Eindeutigkeit
- Chomsky-Normalform
- CYK-Algorithmus

Der CYK-Algorithmus

Karlsruher Institut für Technologie

Satz:
Es gibt einen Algorithmus (den Cocke-Younger-Kasami Algorithmus), der für eine kontextfreie Grammatik G in Chomsky-Normalform und ein Wort $w \in \Sigma^*$ in Zeit $\mathcal{O}(|R| \cdot n^3)$ entscheidet, ob $w \in L(G)$, wobei $n = |w|$ und $|R|$ die Anzahl der Regeln von G ist.

Beweis – Beschreibung des CYK-Algorithmus

- Sei $w = w_1 \dots w_n$.
- Sei $V_{ij} \subseteq V$ so dass $A \in V_{ij}$ genau dann, wenn $A \xrightarrow{*} w_i \dots w_j$.

- Für alle $1 \leq i \leq j \leq n$ berechne die Menge $V_{ij} \subseteq V$.
- Dann ist $w \in L(G)$ genau dann, wenn $S \in V_{1n}$ ist.

- Die Tabelle der V_{ij} wird nach wachsendem $\ell := j - i$ aufgebaut, beginnend mit $\ell = 0$.
- Für $j - i = \ell > 0$ wird die Berechnung von V_{ij} systematisch auf zuvor berechnete V_{ik}, $V_{k+1\,j}$ mit $i \leq k < j$ zurückgeführt.

Bemerkung

- Wir benutzen dynamische Programmierung.

CYK-Algorithmus – Beispiel

$w = 1^* \cup (01)^*$

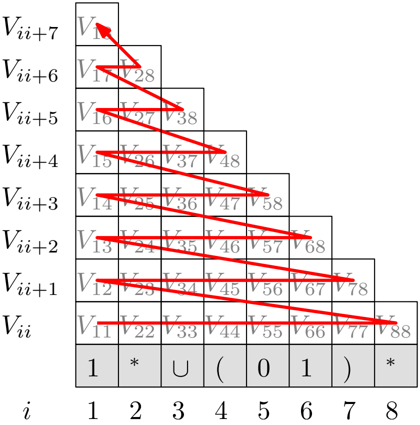
$$
\begin{aligned}
S &\rightarrow SY_* \mid SC_1 \mid \\
& SS \mid Y_(C_2 \mid \\
& e \mid \emptyset \mid 0 \mid 1 \\
C_1 &\rightarrow Y_\cup S \\
C_2 &\rightarrow SY_) \\
Y_* &\rightarrow * \\
Y_\cup &\rightarrow \cup \\
Y_(&\rightarrow (\\
Y_) &\rightarrow)
\end{aligned}
$$

V_{ii+7}	V_{18}							
V_{ii+6}	V_{17}	V_{28}						
V_{ii+5}	V_{16}	V_{27}	V_{38}					
V_{ii+4}	V_{15}	V_{26}	V_{37}	V_{48}				
V_{ii+3}	V_{14}	V_{25}	V_{36}	V_{47}	V_{58}			
V_{ii+2}	V_{13}	V_{24}	V_{35}	V_{46}	V_{57}	V_{68}		
V_{ii+1}	V_{12}	V_{23}	V_{34}	V_{45}	V_{56}	V_{67}	V_{78}	
V_{ii}	V_{11}	V_{22}	V_{33}	V_{44}	V_{55}	V_{66}	V_{77}	V_{88}
	1	*	$\cup$	(	0	1	)	*
i	1	2	3	4	5	6	7	8

CYK-Algorithmus – Beispiel

$w = 1^* \cup (01)^*$

$$
\begin{array}{lcl}
S & \rightarrow & SY_* \mid SC_1 \mid \\
 & & SS \mid Y_(C_2 \mid \\
 & & e \mid \emptyset \mid 0 \mid 1 \\
C_1 & \rightarrow & Y_\cup S \\
C_2 & \rightarrow & SY_) \\
Y_* & \rightarrow & * \\
Y_\cup & \rightarrow & \cup \\
Y_(& \rightarrow & (\\
Y_) & \rightarrow &)
\end{array}
$$



CYK-Algorithmus – Vorgehen

Fall $\ell = 0$:

- Konstruiere die Mengen V_{ii}, d.h. alle $A \in V$ mit $A \xrightarrow{*} w_i$.
- Da G in Chomsky-Normalform ist, gilt $A \xrightarrow{*} w_i$ nur, wenn $(A \to w_i) \in R$.
- Die Berechnung von V_{ii} ist für alle $i \in \{1, \dots n\}$ in $\mathcal{O}(|R|)$ möglich.

CYK-Algorithmus – Beispiel

$w = 1^* \cup (01)^*$

$$
\begin{aligned}
S &\rightarrow SY_* \mid SC_1 \mid \\
&\quad\; SS \mid Y_(C_2 \mid \\
&\quad\; e \mid \emptyset \mid 0 \mid 1 \\
C_1 &\rightarrow Y_\cup S \\
C_2 &\rightarrow SY_) \\
Y_* &\rightarrow * \\
Y_\cup &\rightarrow \cup \\
Y_(&\rightarrow (\\
Y_) &\rightarrow)
\end{aligned}
$$

V_{ii+7}	V_{18}							
V_{ii+6}	V_{17}	V_{28}						
V_{ii+5}	V_{16}	V_{27}	V_{38}					
V_{ii+4}	V_{15}	V_{26}	V_{37}	V_{48}				
V_{ii+3}	V_{14}	V_{25}	V_{36}	V_{47}	V_{58}			
V_{ii+2}	V_{13}	V_{24}	V_{35}	V_{46}	V_{57}	V_{68}		
V_{ii+1}	V_{12}	V_{23}	V_{34}	V_{45}	V_{56}	V_{67}	V_{78}	
V_{ii}	V_{11}	V_{22}	V_{33}	V_{44}	V_{55}	V_{66}	V_{77}	V_{88}
	1	*	$\cup$	(	0	1	)	*
i	1	2	3	4	5	6	7	8

CYK-Algorithmus – Beispiel

$w = 1^* \cup (01)^*$

$$
\begin{aligned}
S &\rightarrow SY_* \mid SC_1 \mid \\
  &\quad\; SS \mid Y_(C_2 \mid \\
  &\quad\; e \mid \emptyset \mid 0 \mid 1 \\
C_1 &\rightarrow Y_\cup S \\
C_2 &\rightarrow SY_) \\
Y_* &\rightarrow * \\
Y_\cup &\rightarrow \cup \\
Y_(&\rightarrow (\\
Y_) &\rightarrow)
\end{aligned}
$$

V_{ii+7}	V_{18}							
V_{ii+6}	V_{17}	V_{28}						
V_{ii+5}	V_{16}	V_{27}	V_{38}					
V_{ii+4}	V_{15}	V_{26}	V_{37}	V_{48}				
V_{ii+3}	V_{14}	V_{25}	V_{36}	V_{47}	V_{58}			
V_{ii+2}	V_{13}	V_{24}	V_{35}	V_{46}	V_{57}	V_{68}		
V_{ii+1}	V_{12}	V_{23}	V_{34}	V_{45}	V_{56}	V_{67}	V_{78}	
V_{ii}	S	Y_*	$Y_\cup$	$Y_($	S	S	$Y_)$	Y_*
	1	$*$	$\cup$	$($	0	1	$)$	$*$
i	1	2	3	4	5	6	7	8

CYK-Algorithmus – Vorgehen

Fall $\ell > 0$:

- Jede Ableitung von $w_i \dots w_j$ muss mit einer Regel der Form

$$A \rightarrow BC$$

beginnen, wobei ein $k \in \{i, \dots, j-1\}$ existiert mit

  - $B \xrightarrow{*} w_i \dots w_k$ und
  - $C \xrightarrow{*} w_{k+1} \dots w_j$.

CYK-Algorithmus – Vorgehen

Verfahren

- Speichere alle Mengen V_{rs} als Arrays der Länge $|V|$, in denen für jedes $A \in V$ markiert ist, ob $A \in V_{rs}$.
- **Berechnung von V_{ij}:**
  Überprüfe für jede Regel $(A \to BC) \in R$ und jedes k, ob

$$\begin{aligned} B &\xrightarrow{*} w_i \dots w_k \\ C &\xrightarrow{*} w_{k+1} \dots w_j \end{aligned}$$

  durch Ansehen der Stelle
  - B im Array zu V_{ik} und
  - C im Array zu $V_{k+1\ j}$.

CYK-Algorithmus – Vorgehen

Karlsruher Institut für Technologie

Verfahren

- Speichere alle Mengen V_{rs} als Arrays der Länge $|V|$, in denen für jedes $A \in V$ markiert ist, ob $A \in V_{rs}$.
- **Berechnung von V_{ij}:**
  Überprüfe für jede Regel $(A \to BC) \in R$ und jedes k, ob

$$
\begin{aligned}
B &\overset{*}{\to} w_i \dots w_k \\
C &\overset{*}{\to} w_{k+1} \dots w_j
\end{aligned}
$$

  durch Ansehen der Stelle
  - B im Array zu V_{ik} und
  - C im Array zu $V_{k+1\ j}$.

Bemerkung

- Dies benötigt Aufwand in $\mathcal{O}(n \cdot |R|)$

CYK-Algorithmus – Beispiel

$w = 1^* \cup (01)^*$

$$
\begin{aligned}
S &\rightarrow SY_* \mid SC_1 \mid \\
& SS \mid Y_(C_2 \mid \\
& e \mid \emptyset \mid 0 \mid 1 \\
C_1 &\rightarrow Y_\cup S \\
C_2 &\rightarrow SY_) \\
Y_* &\rightarrow * \\
Y_\cup &\rightarrow \cup \\
Y_(&\rightarrow (\\
Y_) &\rightarrow)
\end{aligned}
$$

V_{ii+7}	V_{18}							
V_{ii+6}	V_{17}	V_{28}						
V_{ii+5}	V_{16}	V_{27}	V_{38}					
V_{ii+4}	V_{15}	V_{26}	V_{37}	V_{48}				
V_{ii+3}	V_{14}	V_{25}	V_{36}	V_{47}	V_{58}			
V_{ii+2}	V_{13}	V_{24}	V_{35}	V_{46}	V_{57}	V_{68}		
V_{ii+1}	V_{12}	V_{23}	V_{34}	V_{45}	V_{56}	V_{67}	V_{78}	
V_{ii}	S	Y_*	$Y_\cup$	$Y_($	S	S	$Y_)$	Y_*
	1	$*$	$\cup$	$($	0	1	$)$	$*$
i	1	2	3	4	5	6	7	8

CYK-Algorithmus – Beispiel

$w = 1^* \cup (01)^*$

$$
\begin{aligned}
S &\rightarrow SY_* \mid SC_1 \mid \\
& SS \mid Y_(C_2 \mid \\
& e \mid \emptyset \mid 0 \mid 1 \\
C_1 &\rightarrow Y_\cup S \\
C_2 &\rightarrow SY_) \\
Y_* &\rightarrow * \\
Y_\cup &\rightarrow \cup \\
Y_(&\rightarrow (\\
Y_) &\rightarrow)
\end{aligned}
$$

V_{ii+7}	V_{18}							
V_{ii+6}	V_{17}	V_{28}						
V_{ii+5}	V_{16}	V_{27}	V_{38}					
V_{ii+4}	V_{15}	V_{26}	V_{37}	V_{48}				
V_{ii+3}	V_{14}	V_{25}	V_{36}	V_{47}	V_{58}			
V_{ii+2}	V_{13}	V_{24}	V_{35}	V_{46}	V_{57}	V_{68}		
V_{ii+1}	S	$\emptyset$	$\emptyset$	$\emptyset$	S	C_2	$\emptyset$	
V_{ii}	S	Y_*	$Y_\cup$	$Y_($	S	S	$Y_)$	Y_*
	1	$*$	$\cup$	$($	0	1	$)$	$*$
i	1	2	3	4	5	6	7	8

CYK-Algorithmus – Beispiel

$w = 1^* \cup (01)^*$

$$
\begin{array}{rcl}
S & \rightarrow & SY_* \mid SC_1 \mid \\
 & & SS \mid Y_(C_2 \mid \\
 & & e \mid \emptyset \mid 0 \mid 1 \\
C_1 & \rightarrow & Y_\cup S \\
C_2 & \rightarrow & SY_) \\
Y_* & \rightarrow & * \\
Y_\cup & \rightarrow & \cup \\
Y_(& \rightarrow & (\\
Y_) & \rightarrow &)
\end{array}
$$

V_{ii+7}	V_{18}							
V_{ii+6}	V_{17}	V_{28}						
V_{ii+5}	V_{16}	V_{27}	V_{38}					
V_{ii+4}	V_{15}	V_{26}	V_{37}	V_{48}				
V_{ii+3}	V_{14}	V_{25}	V_{36}	V_{47}	V_{58}			
V_{ii+2}	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	C_2	$\emptyset$		
V_{ii+1}	S	$\emptyset$	$\emptyset$	$\emptyset$	S	C_2	$\emptyset$	
V_{ii}	S	Y_*	$Y_\cup$	$Y_($	S	S	$Y_)$	Y_*
	1	$*$	$\cup$	$($	0	1	$)$	$*$
i	1	2	3	4	5	6	7	8

CYK-Algorithmus – Beispiel

$$w = 1^* \cup (01)^*$$

$$
\begin{aligned}
S &\rightarrow SY_* \mid SC_1 \mid \\
& SS \mid Y_(C_2 \mid \\
& e \mid \emptyset \mid 0 \mid 1 \\
C_1 &\rightarrow Y_\cup S \\
C_2 &\rightarrow SY_) \\
Y_* &\rightarrow * \\
Y_\cup &\rightarrow \cup \\
Y_(&\rightarrow (\\
Y_) &\rightarrow)
\end{aligned}
$$

V_{ii+7}	V_{18}							
V_{ii+6}	V_{17}	V_{28}						
V_{ii+5}	V_{16}	V_{27}	V_{38}					
V_{ii+4}	V_{15}	V_{26}	V_{37}	V_{48}				
V_{ii+3}	$\emptyset$	$\emptyset$	$\emptyset$	S	$\emptyset$			
V_{ii+2}	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	C_2	$\emptyset$		
V_{ii+1}	S	$\emptyset$	$\emptyset$	$\emptyset$	S	C_2	$\emptyset$	
V_{ii}	S	Y_*	$Y_\cup$	$Y_($	S	S	$Y_)$	Y_*
	1	*	$\cup$	(	0	1	)	*
i	1	2	3	4	5	6	7	8

CYK-Algorithmus – Beispiel

$w = 1^* \cup (01)^*$

$$
\begin{aligned}
S &\to SY_* \mid SC_1 \mid \\
&\ SS \mid Y_(C_2 \mid \\
&\ e \mid \emptyset \mid 0 \mid 1 \\
C_1 &\to Y_\cup S \\
C_2 &\to SY_) \\
Y_* &\to * \\
Y_\cup &\to \cup \\
Y_(&\to (\\
Y_) &\to)
\end{aligned}
$$

V_{ii+7}	V_{18}							
V_{ii+6}	V_{17}	V_{28}						
V_{ii+5}	V_{16}	V_{27}	V_{38}					
V_{ii+4}	$\emptyset$	$\emptyset$	C_1	S				
V_{ii+3}	$\emptyset$	$\emptyset$	$\emptyset$	S	$\emptyset$			
V_{ii+2}	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	C_2	$\emptyset$		
V_{ii+1}	S	$\emptyset$	$\emptyset$	$\emptyset$	S	C_2	$\emptyset$	
V_{ii}	S	Y_*	$Y_\cup$	$Y_($	S	S	$Y_)$	Y_*
	1	$*$	$\cup$	(	0	1	)	$*$
i	1	2	3	4	5	6	7	8

CYK-Algorithmus – Beispiel

$w = 1^* \cup (01)^*$

$$
\begin{array}{lcl}
S & \to & SY_* \mid SC_1 \mid \\
 & & SS \mid Y_(C_2 \mid \\
 & & e \mid \emptyset \mid 0 \mid 1 \\
C_1 & \to & Y_\cup S \\
C_2 & \to & SY_) \\
Y_* & \to & * \\
Y_\cup & \to & \cup \\
Y_(& \to & (\\
Y_) & \to &)
\end{array}
$$

V_{ii+7}	V_{18}							
V_{ii+6}	V_{17}	V_{28}						
V_{ii+5}	$\emptyset$	$\emptyset$	$\emptyset$					
V_{ii+4}	$\emptyset$	$\emptyset$	C_1	S				
V_{ii+3}	$\emptyset$	$\emptyset$	$\emptyset$	S	$\emptyset$			
V_{ii+2}	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	C_2	$\emptyset$		
V_{ii+1}	S	$\emptyset$	$\emptyset$	$\emptyset$	S	C_2	$\emptyset$	
V_{ii}	S	Y_*	$Y_\cup$	$Y_($	S	S	$Y_)$	Y_*
	1	*	$\cup$	(	0	1	)	*
i	1	2	3	4	5	6	7	8

CYK-Algorithmus – Beispiel

$w = 1^* \cup (01)^*$

$$
\begin{aligned}
S &\rightarrow SY_* \mid SC_1 \mid \\
&\quad\; SS \mid Y_(C_2 \mid \\
&\quad\; e \mid \emptyset \mid 0 \mid 1 \\
C_1 &\rightarrow Y_\cup S \\
C_2 &\rightarrow SY_) \\
Y_* &\rightarrow * \\
Y_\cup &\rightarrow \cup \\
Y_(&\rightarrow (\\
Y_) &\rightarrow)
\end{aligned}
$$

V_{ii+7}	V_{18}							
V_{ii+6}	S	$\emptyset$						
V_{ii+5}	$\emptyset$	$\emptyset$	$\emptyset$					
V_{ii+4}	$\emptyset$	$\emptyset$	C_1	S				
V_{ii+3}	$\emptyset$	$\emptyset$	$\emptyset$	S	$\emptyset$			
V_{ii+2}	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	C_2	$\emptyset$		
V_{ii+1}	S	$\emptyset$	$\emptyset$	$\emptyset$	S	C_2	$\emptyset$	
V_{ii}	S	Y_*	$Y_\cup$	$Y_($	S	S	$Y_)$	Y_*
	1	*	$\cup$	(	0	1	)	*
i	1	2	3	4	5	6	7	8

CYK-Algorithmus – Beispiel

$w = 1^* \cup (01)^*$

$$
\begin{aligned}
S &\rightarrow SY_* \mid SC_1 \mid \\
& SS \mid Y_(C_2 \mid \\
& e \mid \emptyset \mid 0 \mid 1 \\
C_1 &\rightarrow Y_\cup S \\
C_2 &\rightarrow SY_) \\
Y_* &\rightarrow * \\
Y_\cup &\rightarrow \cup \\
Y_(&\rightarrow (\\
Y_) &\rightarrow)
\end{aligned}
$$

V_{ii+7}	S							
V_{ii+6}	S	$\emptyset$						
V_{ii+5}	$\emptyset$	$\emptyset$	$\emptyset$					
V_{ii+4}	$\emptyset$	$\emptyset$	C_1	S				
V_{ii+3}	$\emptyset$	$\emptyset$	$\emptyset$	S	$\emptyset$			
V_{ii+2}	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	C_2	$\emptyset$		
V_{ii+1}	S	$\emptyset$	$\emptyset$	$\emptyset$	S	C_2	$\emptyset$	
V_{ii}	S	Y_*	$Y_\cup$	$Y_($	S	S	$Y_)$	Y_*
	1	*	$\cup$	(	0	1	)	*
i	1	2	3	4	5	6	7	8

Ergebnisse zum Wortproblem

- **Typ-0 Grammatik.** Das Wortproblem ist nicht entscheidbar.
- **Typ-1 Grammatik.** Das Wortproblem ist $\mathcal{NP}$-vollständig.
- **Typ-2 Grammatik.** Das Wortproblem ist in polynomieller Zeit lösbar.
- **Typ-3 Grammatik.** Das Wortproblem ist in linearer Zeit lösbar.