Automatentheorie und Formale Sprachen

für die Studiengänge
- Angewandte Informatik
- Informatik - Technische Systeme

06 Kontextfreie Sprachen

Prof. Dr. David Sabel
Sommersemester 2025

Stand der Folien: 28. Mai 2025

Kontextfreie Sprachen

Zur Erinnerung:

- Kontextfreie Sprachen (CFLs) werden von einer kontextfreien Grammatik (CFG) erzeugt
- Das sind die Typ 2-Grammatiken
- Bedingung: Alle linken Seiten der Produktionen bestehen aus genau einer Variablen, d.h. sie sind von der Form $A \to r$.

Motivation

- Kontextfreie Sprachen sind insbesondere nützlich um Sprachen mit Klammerungen zu beschreiben
- Die Syntax von Programmiersprachen wird meist mit einer kontextfreien Grammatik angegeben

Beispiele:

- $G = (\{E, M, Z, N, D\}, \{+, *, (,)\} \cup \{0, \ldots, 9\}, P, E)$ mit

$$\begin{aligned} P = \{ & E \to M \mid E + M, \\ & M \to Z \mid M * Z, \\ & Z \to N \mid (E), \\ & N \to 1D \mid \ldots \mid 9D, \\ & D \to 0D \mid \ldots \mid 9D \mid \varepsilon\} \end{aligned}$$

- $L = \{a^j b^j \mid j \in \mathbb{N}_0\}$ ist kontextfrei: Produktionen $\{S \to \varepsilon \mid T, T \to aTb \mid ab\}$ mit S als Startsymbol erzeugen L

Normalformen

- Normalformen von Grammatiken fordern eine spezielle Form der Produktionen
- Nützlich, wenn man Grammatiken analysiert oder Algorithmen auf Grammatiken formuliert
- Man muss dann nur diese Form (statt aller erlaubten) von Produktionen betrachten
- Wir betrachten zwei Normalformen
  - Chomsky-Normalform
  - Greibach-Normalform

Die Chomsky-Normalform

Definition (Chomsky-Normalform)

Eine CFG $G = (V, \Sigma, P, S)$ mit $\varepsilon \notin L(G)$ ist in **Chomsky-Normalform**, wenn für jede Produktion $A \to w \in P$ gilt: $w = a \in \Sigma$ oder $w = BC$ mit $B, C \in V$.

Die Chomsky-Normalform

Definition (Chomsky-Normalform)

Eine CFG $G = (V, \Sigma, P, S)$ mit $\varepsilon \notin L(G)$ ist in **Chomsky-Normalform**, wenn für jede Produktion $A \to w \in P$ gilt: $w = a \in \Sigma$ oder $w = BC$ mit $B, C \in V$.

Beispiel:

- Die CFG $G = (\{A\}, \{(,),[,]\}, \{A \to (A) \mid () \mid [A] \mid [] \mid A\,A\}, A)$ ist **nicht** in Chomsky-Normalform (nur die Produktion $A \to A\,A$ passt zum vorgeschriebenen Format).
- Die CFG $G' = (\{A, B, C, D, E, F, G\}, \{(,),[,]\}, P, A)$ mit

$$\begin{aligned} P = \{&A \to BF \mid BC \mid DG \mid DE \mid AA,\\ &B \to (, C \to), D \to [, E \to], F \to AC, G \to AE\} \end{aligned}$$

ist in Chomsky-Normalform (und erzeugt die gleiche Sprache wie G).

Quiz 1

Welche der Grammatiken $G_i = (\{A, B, C, D\}, \{a, b\}, P_i, A)$ sind in Chomsky-Normalform, wobei

$$P_1 = \{A \to BC, A \to a, B \to ABA, B \to b, C \to DD, D \to b\}$$

$$P_2 = \{A \to B, B \to C, C \to CD, C \to a, D \to b\}$$

$$P_3 = \{A \to AB, A \to BC, C \to aDa, D \to bab\}$$

$$P_4 = \{A \to AB, A \to a, B \to CD, C \to BA, C \to a, D \to b\}$$

arsnova.hs-rm.de

6750 1376

Eigenschaften der Chomsky-Normalform

Sei G eine CFG in Chomsky-Normalform, dann gilt:

- Syntaxbäume zu Ableitungen mit G sind immer **Binärbäume**
- Ableitungen eines Worts $w \in L(G)$ bestehen immer genau aus $2 \cdot |w| - 1$ Ableitungsschritten.

Beispiel:

$G' = (\{A, B, C, D, E, F, G\}, \{(,), [,]\}, P, A)$ mit

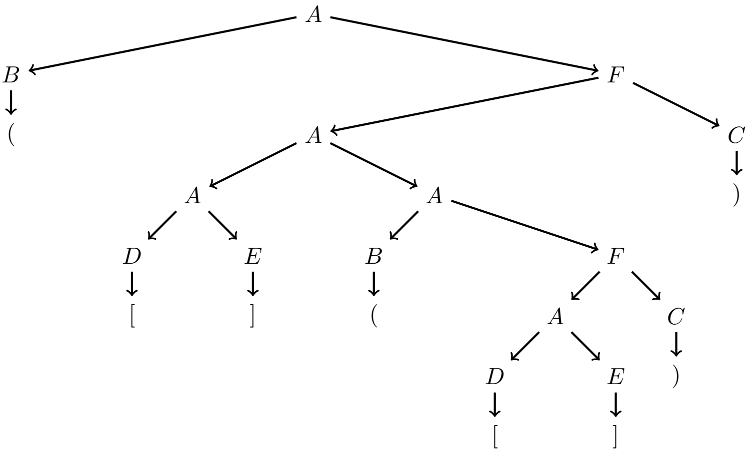
$$\begin{aligned} P = \{&A \to BF \mid BC \mid DG \mid DE \mid AA, \\ &B \to (, C \to), D \to [, E \to], F \to AC, G \to AE\} \end{aligned}$$

Ableitung von $([]([]))$:

$$\begin{aligned} &A \Rightarrow BF \Rightarrow (F \Rightarrow (AC \Rightarrow (AAC \Rightarrow (DEAC \Rightarrow ([EAC \\ &\Rightarrow ([]AC \Rightarrow ([]BFC \Rightarrow ([](FC \Rightarrow ([](ACC \Rightarrow ([](DECC \\ &\Rightarrow ([]([ECC \Rightarrow ([]([]CC \Rightarrow ([]([])C \Rightarrow ([]([])) \end{aligned}$$

Eigenschaften der Chomsky-Normalform (2)

Der Syntaxbaum dazu:



Herstellen der Chomsky-Normalform

Theorem

Für CFGs G mit $\varepsilon \notin L(G)$ kann eine äquivalente CFG in Chomsky-Normalform berechnet werden.

Skizze:

1. Entfernen von ε-Produktionen
2. Entfernen von Einheitsproduktionen $A \to B$
3. Sharen aller Terminale a in rechten Seiten, die nicht nur aus a bestehen durch neue Produktionen $A \to a$
4. Alle Produktionen $A \to B_1 \cdots B_m$ mit $m > 2$ in mehrere zerlegen:
   $A \to B_1 C_1, C_1 \to B_2 C_3, \ldots, C_{m-2} \to B_{m-1} B_m$

Die genauen Details können in der Literatur gefunden werden.

Greibach-Normalform

Definition (Greibach-Normalform)

Ein CFG $G = (V, \Sigma, P, S)$ mit $\varepsilon \notin L(G)$ ist in *Greibach-Normalform*, falls alle Produktionen in P von der Form $A \to aB_1B_2 \ldots B_j$ mit $j \geq 0$, $A, B_1, \ldots, B_j \in V$ und $a \in \Sigma$ sind.

Bemerkungen:

- benannt nach Sheila A. Greibach
- Reguläre Grammatiken sind Spezialfall der Greibach-NF:
  Dort ist nur $j = 0$ oder $j = 1$ erlaubt.
- Die Greibach-Normalform wird u.a. verwendet, um zu zeigen, dass Kontextfreie Sprachen genau von den nichtdeterministischen Kellerautomaten erkannt werden (später).

Greibach-Normalform kann hergestellt werden

Satz

Zu jeder CFG G mit $\varepsilon \notin L(G)$ gibt es eine CFG G' in Greibach-Normalform, sodass $L(G) = L(G')$ gilt.

Beweis und Algorithmus: siehe Literatur

Quiz 2

Welche der Grammatiken $G_i = (\{A, B, C, D\}, \{a, b\}, P_i, A)$ sind in Greibach-Normalform, wobei

$$P_1 = \{A \to BC, A \to a, B \to Aa, B \to b, C \to DD, D \to b\}$$

$$P_2 = \{A \to aBCD, B \to bC, C \to aDD, C \to a, D \to b\}$$

$$P_3 = \{A \to AB, A \to BC, C \to aDa, D \to bab\}$$

$$P_4 = \{A \to abaA, A \to a, B \to bbCD, C \to aB, C \to a, D \to b\}$$

arsnova.hs-rm.de

6750 1376

Abschlusseigenschaften der kontextfreien Sprachen

Theorem

Die kontextfreien Sprachen sind abgeschlossen unter Vereinigung, Produkt und Kleeneschem Abschluss.

Beweis:

- Seien L_1,L_2 CFLs und $G_i = (V_i, \Sigma_i, P_i, S_i)$ CFGs mit $L(G_i) = L_i$ für $i = 1, 2$. O.B.d.A. sei $V_1 \cap V_2 = \emptyset$.
- Seien S, S' neue Variablen ($\{S, S'\} \cap (V_1 \cup V_2) = \emptyset$).

Vereinigung: Sei

$$G_\cup = (V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P_1 \cup P_2 \cup \{S \to S_1 \mid S_2\}, S)$$

Dann gilt: $L(G_\cup) = L(G_1) \cup L(G_2) = L_1 \cup L_2$.

Abschlusseigenschaften der kontextfreien Sprachen (2)

Beweis (Fortsetzung):

Produkt: Sei

$$G_\circ = (V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P_1 \cup P_2 \cup \{S \to S_1 S_2\}, S)$$

Dann gilt $L(G_\circ) = L(G_1)L(G_2) = L_1 L_2$.

Abschlusseigenschaften der kontextfreien Sprachen (3)

Beweis (Fortsetzung):

Kleenescher Abschluss: Sei

$$G_{1,*} = (V_1 \cup \{S', S\}, \Sigma, P', S')$$

mit

$$P' = (P \setminus \{S_1 \to \varepsilon\}) \cup \{S' \to \varepsilon, S' \to S, S \to SS, S \to S_1\}$$

Dann gilt $L(G_{1,*}) = L(G_1)^*$.

Nichtkontextfreie Sprache

Satz

Die Sprache $L = \{a^l b^l c^l \mid l \in \mathbb{N}_0\}$ ist nicht kontextfrei.

Ein Beweis hierfür sehen wir später mit dem Pumping-Lemma
für Kontextfreie Sprachen.

Wir verwenden die Sprache und die Eigenschaft jedoch im Folgenden.

Abschlusseigenschaften der kontextfreien Sprachen (4)

Theorem

Die kontextfreien Sprachen sind **nicht abgeschlossen** unter Schnitt- und Komplementbildung.

Beweis:

Schnittbildung:

- Sei $L_1 = \{a^n b^m c^m \mid m, n \in \mathbb{N}_0\}$ und sei $L_2 = \{a^m b^m c^n \mid m, n \in \mathbb{N}_0\}$
- $G_1 = (\{A, D, S\}, \{a, b, c\}, \{S \to AD, A \to \varepsilon \mid aA, D \to bDc \mid \varepsilon\}, S)$
  $G_2 = (\{C, D, S\}, \{a, b, c\}, \{S \to DC, C \to cC \mid \varepsilon, D \to aDb \mid \varepsilon\}, S)$
  Für i=1,2: $L(G_i) = L_i$, daher: L_1 und L_2 sind beide kontextfrei.
- $L_1 \cap L_2 = \{a^n b^n c^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei.
- Daher sind die CFLs nicht abgeschlossen bezüglich Schnittbildung.

Abschlusseigenschaften der kontextfreien Sprachen (5)

Beweis (Fortsetzung):

Komplement:

- Widerspruchsbeweis: Wir nehmen an, es gilt:
$$L \text{ ist CFL } \implies \overline{L} \text{ ist CFL}$$
- Seien L_1, L_2 CFLs. Dann ist auch $\overline{\overline{L_1} \cup \overline{L_2}}$ CFL
(da CFLs abgeschlossen bez. $\bar{\cdot}$ und $\cup$)
- Aber: $\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$. Widerspruch!

WIDERLEGEN DER KONTEXTFREIHEIT

- Pumping-Lemma für Kontextfreie Sprachen

Widerlegen der Kontextfreiheit

Wir lernen eine Methode kennen zum Widerlegen der Kontextfreiheit:

- Pumping-Lemma für kontextfreie Sprachen

Es gibt weitere (allgemeinere) Formulierungen, z.B.

- Ogdens-Lemma (benannt nach William F. Ogden)
- Interchange-Lemma

Einschub: Binärbäume

Binärbaum: Baum, wobei jeder Knoten 0 oder 2 Kinder hat

Lemma

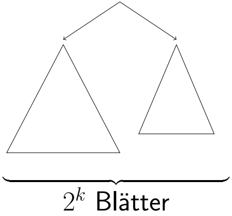
Sei B ein Binärbaum mit $\geq 2^k$ Blättern. Dann hat B einen Pfad der Länge $\geq k$.

Beweis durch Induktion über k:

$k = 0$:

- Ein Baum mit $2^k = 2^0 = 1$ Blättern besteht genau aus diesem Blatt und hat einen Pfad der Länge ≥ 0.

$k > 0$:



- Einer der beiden Teilbäume unter der Wurzel hat $\geq 2^{k-1}$ Blätter.
- Per Induktionsannahme hat dieser einen Pfad der Länge $\geq k - 1$.
- Daher hat der gesamte Baum einen Pfad der Länge $\geq k$.

Pumping-Lemma für CFLs: Bemerkungen

- **Erinnerung:** Pumping-Lemma für reguläre Sprachen:
  Jede reguläre Sprache erfüllt die Pumping-Eigenschaft.

Pumping-Lemma für CFLs: Bemerkungen

- **Erinnerung:** Pumping-Lemma für reguläre Sprachen:
  Jede reguläre Sprache erfüllt die Pumping-Eigenschaft.
- Analog: Pumping-Lemma für kontextfreie Sprachen:
  Jede kontextfreie Sprache erfüllt die Pumping-Eigenschaft für kontextfreie Sprachen

Pumping-Lemma für CFLs: Bemerkungen

- **Erinnerung:** Pumping-Lemma für reguläre Sprachen:
  Jede reguläre Sprache erfüllt die Pumping-Eigenschaft.
- Analog: Pumping-Lemma für kontextfreie Sprachen:
  Jede kontextfreie Sprache erfüllt die Pumping-Eigenschaft für kontextfreie Sprachen
- Kann vorallem zum Widerlegen benutzt werden:
  Sprache **verletzt** die Pumping-Eigenschaft für CFLs
  $\Longrightarrow$ Sprache ist **nicht kontextfrei**

Pumping-Lemma für CFLs: Bemerkungen

- **Erinnerung:** Pumping-Lemma für reguläre Sprachen:
  Jede reguläre Sprache erfüllt die Pumping-Eigenschaft.
- Analog: Pumping-Lemma für kontextfreie Sprachen:
  Jede kontextfreie Sprache erfüllt die Pumping-Eigenschaft für kontextfreie Sprachen
- Kann vorallem zum Widerlegen benutzt werden:
  Sprache **verletzt** die Pumping-Eigenschaft für CFLs
  $\Longrightarrow$ Sprache ist **nicht kontextfrei**
- Pumping-Eigenschaft bei regulären Sprachen, informell:
  Man kann Wörter an einer Stelle aufpumpen und verbleibt in der Sprache
  ($uv^iw \in L$ für alle $i \in \mathbb{N}_0$)

Pumping-Lemma für CFLs: Bemerkungen

- **Erinnerung:** Pumping-Lemma für reguläre Sprachen:
  Jede reguläre Sprache erfüllt die Pumping-Eigenschaft.
- Analog: Pumping-Lemma für kontextfreie Sprachen:
  Jede kontextfreie Sprache erfüllt die Pumping-Eigenschaft für kontextfreie Sprachen
- Kann vorallem zum Widerlegen benutzt werden:
  Sprache **verletzt** die Pumping-Eigenschaft für CFLs
  $\Longrightarrow$ Sprache ist **nicht kontextfrei**
- Pumping-Eigenschaft bei regulären Sprachen, informell:
  Man kann Wörter an einer Stelle aufpumpen und verbleibt in der Sprache
  ($uv^iw \in L$ für alle $i \in \mathbb{N}_0$)
- Pumping-Eigenschaft bei kontextfreien Sprachen, informell:
  Man kann Wörter an ***zwei*** *Stellen gleichzeitig aufpumpen und verbleibt in der Sprache*
  ($uv^iwx^iy \in L$ für alle $i \in \mathbb{N}_0$)

Das Pumping-Lemma für CFLs

Lemma (Pumping-Lemma für CFLs)

Sei L eine kontextfreie Sprache. Dann gibt es eine Zahl $n \in \mathbb{N}$, sodass jedes Wort $z \in L$, das Mindestlänge n hat (d. h. $|z| \geq n$), als $z = uvwxy$ geschrieben werden kann, so dass gilt:

- $|vx| \geq 1$
- $|vwx| \leq n$
- für alle $i \geq 0$: $uv^iwx^iy \in L$.

Beweis des Pumping-Lemmas (1)

Behauptung: Für jede CFL L gibt es $n \in \mathbb{N}$, sodass jedes Wort $z \in L$ mit $|z| \geq n$ als $z = uvwxy$ geschrieben werden kann mit

- $|vx| \geq 1$
- $|vwx| \leq n$
- für alle $i \geq 0$: $uv^iwx^iy \in L$

Beweis:

- Sei $G = (V, \Sigma, P, S)$ CFG in Chomsky-NF mit $L(G) = L \setminus \{\varepsilon\}$

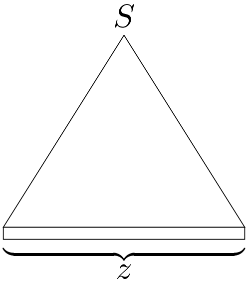
Beweis des Pumping-Lemmas (1)

Behauptung: Für jede CFL L gibt es $n \in \mathbb{N}$, sodass jedes Wort $z \in L$ mit $|z| \geq n$ als $z = uvwxy$ geschrieben werden kann mit

- $|vx| \geq 1$
- $|vwx| \leq n$
- für alle $i \geq 0$: $uv^iwx^iy \in L$

Beweis:

- Sei $G = (V, \Sigma, P, S)$ CFG in Chomsky-NF mit $L(G) = L \setminus \{\varepsilon\}$
- Sei $z \in L(G)$ mit $|z| \geq 2^{|V|} = n$: Ableitung & Syntaxbaum dazu



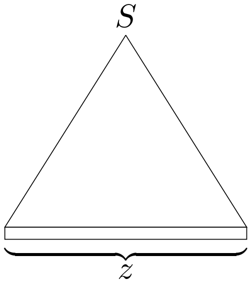
Beweis des Pumping-Lemmas (1)

Behauptung: Für jede CFL L gibt es $n \in \mathbb{N}$, sodass jedes Wort $z \in L$ mit $|z| \geq n$ als $z = uvwxy$ geschrieben werden kann mit

- $|vx| \geq 1$
- $|vwx| \leq n$
- für alle $i \geq 0$: $uv^iwx^iy \in L$

Beweis:

- Sei $G = (V, \Sigma, P, S)$ CFG in Chomsky-NF mit $L(G) = L \setminus \{\varepsilon\}$
- Sei $z \in L(G)$ mit $|z| \geq 2^{|V|} = n$: Ableitung & Syntaxbaum dazu (wenn es solche Wörter nicht gibt, dann gilt die Behauptung)



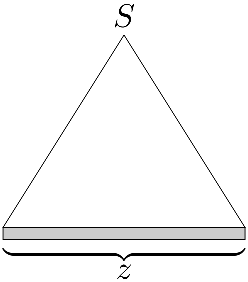
Beweis des Pumping-Lemmas (1)

Behauptung: Für jede CFL L gibt es $n \in \mathbb{N}$, sodass jedes Wort $z \in L$ mit $|z| \geq n$ als $z = uvwxy$ geschrieben werden kann mit

- $|vx| \geq 1$
- $|vwx| \leq n$
- für alle $i \geq 0$: $uv^iwx^iy \in L$

Beweis:

- Sei $G = (V, \Sigma, P, S)$ CFG in Chomsky-NF mit $L(G) = L \setminus \{\varepsilon\}$
- Sei $z \in L(G)$ mit $|z| \geq 2^{|V|} = n$: Ableitung & Syntaxbaum dazu (wenn es solche Wörter nicht gibt, dann gilt die Behauptung)
- Da G in Chomsky-NF: Syntaxbaum ist binärer Baum (ohne letzte Schicht für Produktionen $A \to a$)



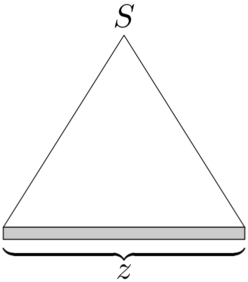
Beweis des Pumping-Lemmas (1)

Behauptung: Für jede CFL L gibt es $n \in \mathbb{N}$, sodass jedes Wort $z \in L$ mit $|z| \geq n$ als $z = uvwxy$ geschrieben werden kann mit

- $|vx| \geq 1$
- $|vwx| \leq n$
- für alle $i \geq 0$: $uv^iwx^iy \in L$

Beweis:

- Sei $G = (V, \Sigma, P, S)$ CFG in Chomsky-NF mit $L(G) = L \setminus \{\varepsilon\}$
- Sei $z \in L(G)$ mit $|z| \geq 2^{|V|} = n$: Ableitung & Syntaxbaum dazu (wenn es solche Wörter nicht gibt, dann gilt die Behauptung)
- Da G in Chomsky-NF: Syntaxbaum ist binärer Baum (ohne letzte Schicht für Produktionen $A \to a$)
- Baum ohne letzte Schicht hat $|z| \geq 2^{|V|}$ Blätter.



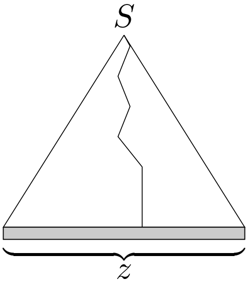
Beweis des Pumping-Lemmas (1)

Behauptung: Für jede CFL L gibt es $n \in \mathbb{N}$, sodass jedes Wort $z \in L$ mit $|z| \geq n$ als $z = uvwxy$ geschrieben werden kann mit

- $|vx| \geq 1$
- $|vwx| \leq n$
- für alle $i \geq 0$: $uv^iwx^iy \in L$

Beweis:

- Sei $G = (V, \Sigma, P, S)$ CFG in Chomsky-NF mit $L(G) = L \setminus \{\varepsilon\}$
- Sei $z \in L(G)$ mit $|z| \geq 2^{|V|} = n$: Ableitung & Syntaxbaum dazu (wenn es solche Wörter nicht gibt, dann gilt die Behauptung)
- Da G in Chomsky-NF: Syntaxbaum ist binärer Baum (ohne letzte Schicht für Produktionen $A \to a$)
- Baum ohne letzte Schicht hat $|z| \geq 2^{|V|}$ Blätter.
- Daher: Es gibt Pfad von Wurzel zum Blatt der Länge $\geq |V|$ mit $\geq |V| + 1$ Knoten, die mit Variablen markiert sind.

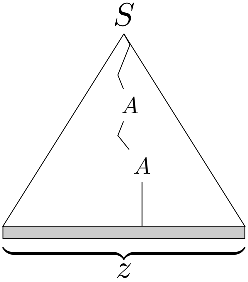


Beweis des Pumping-Lemmas (2)

Beh.: Für jede CFL L gibt es $n \in \mathbb{N}$, sodass jedes Wort $z \in L$ mit $|z| \geq n$ als $z = uvwxy$ geschrieben werden kann mit

- $|vx| \geq 1$
- $|vwx| \leq n$
- für alle $i \geq 0$: $uv^iwx^iy \in L$

Beweis:



- ...
- Da es nur $|V|$ Variablen gibt, kommt mindestens eine Variable mehrfach auf diesem Pfad vor.
- Wähle die Vorkommen der Variablen so, dass das zweite Vorkommen von unten gesehen am tiefsten ist. Sei A die Variable.

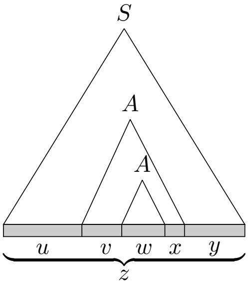
Beweis des Pumping-Lemmas (3)

Beh.: Für jede CFL L gibt es $n \in \mathbb{N}$, sodass jedes Wort $z \in L$ mit $|z| \geq n$ als $z = uvwxy$ geschrieben werden kann mit

- $|vx| \geq 1$
- $|vwx| \leq n$
- für alle $i \geq 0$: $uv^iwx^iy \in L$

Beweis:

- . . .
- Betrachte die Teilbäume, die jeweils A als Wurzel haben.
- Sie entsprechen Ableitungen von Teilworten von z
- Der Teilbaum mit dem unteren A als Wurzel erzeugt ein Teilwort des Teilbaums mit dem oberen A als Wurzel. D.h. $z = uvwxy$, wobei vwx vom oberen A und w vom unteren A erzeugt wird.
- Es gilt $|w| \geq 1$, da Variablen einer Grammatik in Chomsky-Normalform nur Wörter mit Länge ≥ 1 herleiten
- Das Wort vwx muss echt länger sein als w, da das obere A über dem unteren A steht. Daher folgt $|v| \geq 1$ und/oder $|x| \geq 1$ und $|vx| \geq 1$.



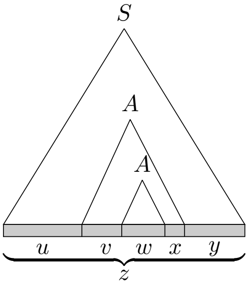
Beweis des Pumping-Lemmas (4)

Beh.: Für jede CFL L gibt es $n \in \mathbb{N}$, sodass jedes Wort $z \in L$ mit $|z| \geq n$ als $z = uvwxy$ geschrieben werden kann mit

- $|vx| \geq 1$
- $|vwx| \leq n$
- für alle $i \geq 0$: $uv^iwx^iy \in L$

Beweis:

- ...
- Da wir das tiefste Vorkommen der wiederholten Variable gewählt haben, kann der Pfad vom oberen A bis zur Blattebene nur aus $\leq |V| + 1$ Knoten bestehen und Länge $\leq |V|$ haben
- Daraus folgt: $|vwx| \leq 2^{|V|} = n$
- Aus dem Baum folgt: $A \Rightarrow^* w$ und $A \Rightarrow^* vAx$ und daher kann man auch $A \Rightarrow^* v^iwx^i$ für alle $i \in \mathbb{N}_0$ ableiten
- Schließlich folgt daraus $S \Rightarrow^* uv^iwx^iy$ für alle $i \in \mathbb{N}_0$.



Pumping-Lemma: Illustrationen

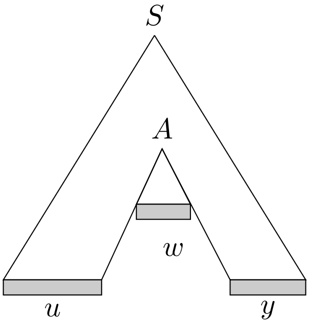


Illustration für uv^0wx^0y

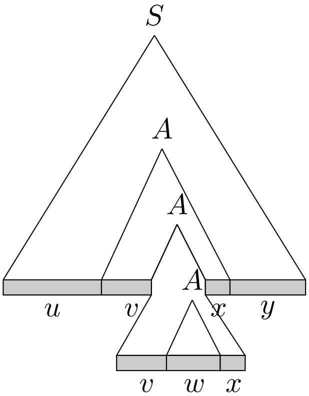


Illustration für uv^2wx^2y

Verwendung des Pumping-Lemma

- Die Pumping-Eigenschaft ist eine **notwendige** aber **keine hinreichende** Bedingung für CFLs.
- Daher kann das Pumping-Lemma **nicht** verwendet werden, um Kontextfreiheit zu zeigen.
- Aber: Es kann verwendet werden, um **Kontextfreiheit zu widerlegen**

Pumping-Lemma zum Widerlegen der Kontextfreiheit

Formulierung des Pumping-Lemmas für CFGs zum Widerlegen der Kontextfreiheit

Sei L eine formale Sprache für die gilt:

> Für jede Zahl $n \in \mathbb{N}$ gibt es ein Wort $z \in L$, das Mindestlänge n hat (d. h. $|z| \geq n$), und für jede Zerlegung $z = uvwxy$ mit $|vwx| \leq n$ und $|vx| \geq 1$, gibt es ein $i \geq 0$, sodass $uv^iwx^iy \notin L$.

Dann ist L nicht kontextfrei.

Beweis:

> Umformung der negierten prädikatenlogischen Formel (siehe Skript), die sich aus dem Pumping-Lemma ergibt.

Pumping-Lemma als Spiel

Sei L die formale Sprache.

1. Der **Gegner** wählt die Zahl $n \in \mathbb{N}$.
2. **Wir** wählen das Wort $z \in L$ mit $|z| \geq n$.
3. Der **Gegner** wählt die Zerlegung
   $z = uvwxy$ mit $|vx| \geq 1$ und $|vwx| \leq n$
4. **Wir** gewinnen das Spiel, wenn wir ein $i \geq 0$ angeben können, sodass $uv^iwx^iy \notin L$.

Wenn wir **für jede Wahl des Gegners** das Spiel gewinnen können,
dann haben wir gezeigt, dass L nicht kontextfrei ist.

Beispiele

Satz

Die Sprache $L = \{a^l b^l c^l \mid l \in \mathbb{N}_0\}$ ist nicht kontextfrei.

Beweis:

- Gegner wählt $n \in \mathbb{N}$

Beispiele

Satz

Die Sprache $L = \{a^l b^l c^l \mid l \in \mathbb{N}_0\}$ ist nicht kontextfrei.

Beweis:

- Gegner wählt $n \in \mathbb{N}$
- Wir wählen $z = a^n b^n c^n$.

Beispiele

Satz

Die Sprache $L = \{a^l b^l c^l \mid l \in \mathbb{N}_0\}$ ist nicht kontextfrei.

Beweis:

- Gegner wählt $n \in \mathbb{N}$
- Wir wählen $z = a^n b^n c^n$.
- Gegner wählt Zerlegung $z = uvwxy$ mit $|vx| \geq 1$ und $|vwx| \leq n$

Beispiele

Satz

Die Sprache $L = \{a^l b^l c^l \mid l \in \mathbb{N}_0\}$ ist nicht kontextfrei.

Beweis:

- Gegner wählt $n \in \mathbb{N}$
- Wir wählen $z = a^n b^n c^n$.
- Gegner wählt Zerlegung $z = uvwxy$ mit $|vx| \geq 1$ und $|vwx| \leq n$
- Fall 1: vwx ist von der Form $a^i b^j$, $i + j \leq n$
  Da $|vx| \geq 1$, gilt $\#_a(vx) \geq 1$ oder $\#_b(vx) \geq 1$, aber $\#_c(vx) = 0$
  Damit folgt $uv^0wx^0y \notin L$

Beispiele

Satz

Die Sprache $L = \{a^l b^l c^l \mid l \in \mathbb{N}_0\}$ ist nicht kontextfrei.

Beweis:

- Gegner wählt $n \in \mathbb{N}$
- Wir wählen $z = a^n b^n c^n$.
- Gegner wählt Zerlegung $z = uvwxy$ mit $|vx| \geq 1$ und $|vwx| \leq n$
- Fall 1: vwx ist von der Form $a^i b^j$, $i + j \leq n$
  Da $|vx| \geq 1$, gilt $\#_a(vx) \geq 1$ oder $\#_b(vx) \geq 1$, aber $\#_c(vx) = 0$
  Damit folgt $uv^0wx^0y \notin L$
- Fall 2: vwx ist von der Form $b^i c^j$, $i + j \leq n$
  Da $|vx| \geq 1$, gilt $\#_b(vx) \geq 1$ oder $\#_c(vx) \geq 1$, aber $\#_a(vx) = 0$
  Damit folgt $uv^0wx^0y \notin L$

Beispiele

Satz

Die Sprache $L = \{a^l b^l c^l \mid l \in \mathbb{N}_0\}$ ist nicht kontextfrei.

Beweis:

- Gegner wählt $n \in \mathbb{N}$
- Wir wählen $z = a^n b^n c^n$.
- Gegner wählt Zerlegung $z = uvwxy$ mit $|vx| \geq 1$ und $|vwx| \leq n$
- Fall 1: vwx ist von der Form $a^i b^j$, $i + j \leq n$
  Da $|vx| \geq 1$, gilt $\#_a(vx) \geq 1$ oder $\#_b(vx) \geq 1$, aber $\#_c(vx) = 0$
  Damit folgt $uv^0wx^0y \notin L$
- Fall 2: vwx ist von der Form $b^i c^j$, $i + j \leq n$
  Da $|vx| \geq 1$, gilt $\#_b(vx) \geq 1$ oder $\#_c(vx) \geq 1$, aber $\#_a(vx) = 0$
  Damit folgt $uv^0wx^0y \notin L$
- Andere Fälle sind nicht möglich!

□

Beispiele (2)

Satz

Die Sprache $L = \{a^i b^j c^i d^j \mid i,j \in \mathbb{N}\}$ ist nicht kontextfrei.

Beweis:

- Gegner wählt $n \in \mathbb{N}$.
- Wir wählen $z = a^n b^n c^n d^n$.
- Gegner wählt Zerlegung $z = uvwxy$ mit $|vx| \geq 1$ und $|vwx| \leq n$
- 1.Fall: $vwx = a^i b^j$ mit $i + j \leq n$. Da $|vx| \geq 1$, gilt $\#_a(vx) + \#_b(vx) \geq 1$ und $uv^0wx^0y = uwy = a^{i'}b^{j'}c^n d^n$ und $i' < n$ und/oder $j' < n$, d.h. $uwy \notin L$.
- 2.Fall: $vwx = b^i c^j$ mit $i + j \leq n$. Da $|vx| \geq 1$, gilt $\#_b(vx) + \#_c(vx) \geq 1$ und $uv^0wx^0y = uwy = a^n b^{i'} c^{j'} d^n$ und $i' < n$ und/oder $j' < n$, d.h. $uwy \notin L$
- 3.Fall: $vwx = c^i d^j$ mit $i + j \leq n$. Da $|vx| \geq 1$, gilt $\#_c(vx) + \#_d(vx) \geq 1$ und $uv^0wx^0y = uwy = a^n b^n c^{i'} d^{j'}$ und $i' < n$ und/oder $j' < n$, d.h. $uwy \notin L$.

Unäres Alphabet

Satz

Sei L eine formale Sprache über einem unären Alphabet (d.h. $|\Sigma| = 1$). Dann ist L genau dann regulär, wenn L kontextfrei ist.

Beweis:

- Wenn L regulär ist, dann ist L auch kontextfrei.
- Rückrichtung: Siehe Skript
  (Beweis verwendet die Pumping-Eigenschaft für CFLs und konstruiert daraus eine Vereinigung von regulären Sprachen)

Beispiele

Satz

Die Sprachen

$$\begin{aligned} L_1 &= \{a^p \mid p \text{ ist eine Primzahl}\} \\ L_2 &= \{a^n \mid n \text{ ist keine Primzahl}\} \\ L_3 &= \{a^n \mid n \text{ ist Quadratzahl}\} \\ L_4 &= \{a^{2^n} \mid n \in \mathbb{N}_0\} \end{aligned}$$

sind allesamt nicht kontextfrei.

Beweis: Wir haben für alle 4 Sprachen gezeigt, dass sie nicht regulär sind. Da sie alle über einem unären Alphabet definiert sind, sind sie auch nicht kontextfrei.

WORTPROBLEM FÜR KONTEXTFREIE SPRACHEN

- Motivation
- Der CYK-Algorithmus

Einleitung zum Wortproblem

- Wir wissen bereits: Das Wortproblem für Typ 1-Sprachen ist entscheidbar
- Daher ist auch das Wortproblem für Typ 2-Sprachen entscheidbar
- Aber: Der Algorithmus für Typ 1-Sprachen benötigt exponentiell viele Schritte
- Nun sehen wir einen Polynomialzeitalgorithmus
- Die wesentliche Entwurfsmethode dabei ist dynamische Programmierung

Effizientes Lösen des Wortproblems für CFLs

- Algorithmus für Typ 1-Sprachen hat exponentielle Laufzeit
- Algorithmus von Cocke, Younger und Kasami für CFLs
- Veröffentlicht in den 1960er Jahren
- Kurz: CYK-Algorithmus

Idee des CYK-Algorithmus

- Eingabe:
  - CFG $G = (V, \Sigma, P, S)$ in Chomsky-Normalform:

    Wiederholung / Bemerkung: G (mit $\varepsilon \notin L(G)$) ist in **Chomsky-Normalform**, wenn für jede Produktion $A \to w \in P$ gilt: $w = a \in \Sigma$ oder $w = BC$ mit $B, C \in V$.

    Satz: Für jede CFG G mit $\varepsilon \notin L(G)$ kann sprachäquivalente Chomsky-NF berechnet werden.
  - Wort $w \in \Sigma^*$
- Ausgabe:
  - ja, wenn $w \in L(G)$
  - nein, wenn $w \notin L(G)$
- Grundidee des Algorithmus:
  (Rekursiver) Test, ob Variable A ein Wort u erzeugt.
- Verwende Test zum Prüfen, ob S das Wort w erzeugt.

Idee des CYK-Algorithmus (2)

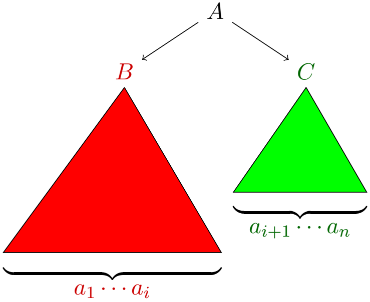
Prüfe, ob $A \in V$ ein Wort $u = a_1 \cdots a_n$ erzeugt:

- Wenn $u = a \in \Sigma$, dann prüfe ob $A \to a \in P$

Idee des CYK-Algorithmus (2)

Prüfe, ob $A \in V$ ein Wort $u = a_1 \cdots a_n$ erzeugt:

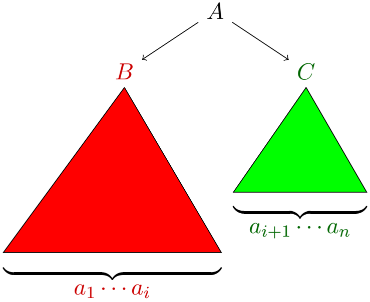
- Wenn $u = a \in \Sigma$, dann prüfe ob $A \to a \in P$
- Anderenfalls ($|u| > 1$) kann u nur erzeugt werden, wenn:
  - Es gibt Produktion $A \to BC \in P$
  - Es gibt Index $1 \leq i < n$, sodass:
    - B erzeugt $a_1 \cdots a_i$ und
    - C erzeugt $a_{i+1} \cdots a_n$



Idee des CYK-Algorithmus (2)

Prüfe, ob $A \in V$ ein Wort $u = a_1 \cdots a_n$ erzeugt:

- Wenn $u = a \in \Sigma$, dann prüfe ob $A \to a \in P$
- Anderenfalls ($|u| > 1$) kann u nur erzeugt werden, wenn:
  - Es gibt Produktion $A \to BC \in P$
  - Es gibt Index $1 \leq i < n$, sodass:
    - B erzeugt $a_1 \cdots a_i$ und
    - C erzeugt $a_{i+1} \cdots a_n$



- Daher prüfe für alle $A \to BC \in P$ und alle i mit $1 \leq i < n$ rekursiv, ob B das Wort $w = a_1 \cdots a_i$ und C das Wort $a_{i+1} \cdots a_n$ erzeugt.

Beispiel

Betrachte die Grammatik mit den Produktionen:

$$S \to AB \mid BA,$$
$$A \to AA \mid AB \mid a,$$
$$B \to BB \mid b$$

und das Wort $bbbaab$.

S erzeugt $bbbaab$, denn $S \to BA$ und

- B erzeugt bbb, denn $B \to BB$ und
  - B erzeugt bb, denn $B \to BB$ und $B \to b$ und $B \to b$
  - B erzeugt b, denn $B \to b$
- A erzeugt aab, denn $A \to AB$ und
  - A erzeugt aa, denn $A \to AA$ und
    - A erzeugt a, denn $A \to a$ und
    - A erzeugt a, denn $A \to a$
  - B erzeugt b, denn $B \to b$

Beispiel

Betrachte die Grammatik mit den Produktionen:

$$S \to AB \mid BA,$$
$$A \to AA \mid AB \mid a,$$
$$B \to BB \mid b$$

und das Wort $bbbaab$.

Bevor der rekursive Algorithmus diesen richtigen Pfad findet, sucht er einige erfolglose ab, z.B. für $S \to AB$

- Prüfe, ob A erzeugt b und B erzeugt $bbab$ gilt.
- Prüfe, ob A erzeugt bb und B erzeugt bab gilt.
- Prüfe, ob A erzeugt bbb und B erzeugt ab gilt.
- Prüfe, ob A erzeugt $bbba$ und B erzeugt b gilt.

S erzeugt $bbbaab$, denn $S \to BA$ und

- B erzeugt bbb, denn $B \to BB$ und
  - B erzeugt bb, denn $B \to BB$ und $B \to b$ und $B \to b$
  - B erzeugt b, denn $B \to b$
- A erzeugt aab, denn $A \to AB$ und
  - A erzeugt aa, denn $A \to AA$ und
    - A erzeugt a, denn $A \to a$ und
    - A erzeugt a, denn $A \to a$
  - B erzeugt b, denn $B \to b$

Naives Suchen wiederholt dieselben Tests, z.B. ob „A erzeugt b“ gilt.

Idee des CYK-Algorithmus (3)

- Effizienz: Statt Rekursion verwende dynamische Programmierung
- Algorithmus berechnet Menge $V(i,j) \subseteq V$, sodass

$$V(i,j) = \{A \in V \mid A \Rightarrow^* a_i \cdots a_{i+j-1}\}$$

„$V(i,j)$ enthält alle Nichtterminale $A \in V$, die $a_i \cdots a_{i+j-1}$ (=Teilwort von u ab Position i mit Länge j) erzeugen“

- Berechnung der $V(i,j)$:
  - Starte mit $V(i,1) = \{A \mid A \to a_i \in P\}$.
  - Berechne $V(i,j)$ mit ansteigender Länge j.

    Für $j > 1$ gilt:

$$A \in V(i,j) \text{ g.d.w.}$$
$$A \to BC \in P \text{ und } B \in V(i,k),\ C \in V(i+k, j-k)$$

Berechnung: Für festes (i,j) betrachte alle k mit $k = 1, 2, \ldots, j-1$

- Finaler Schritt: Prüfe, ob $S \in V(1,n)$ ist.

Algorithmus 4: CYK-Algorithmus

```
Eingabe: CFG $G = (V, \Sigma, P, S)$ in Chomsky-Normalform und Wort $w = a_1 \cdots a_n \in \Sigma^*$
Ausgabe: Ja, wenn $w \in L(G)$ und Nein, wenn $w \notin L(G)$
Beginn
    für $i = 1$ bis $n$ tue
        $V(i,1) = \{A \in V \mid A \to a_i \in P\}$
    für $j = 2$ bis $n$ tue
        für $i = 1$ bis $n + 1 - j$ tue
            $V(i,j) = \emptyset$;
            für $k = 1$ bis $j - 1$ tue
                $V(i,j) = V(i,j) \cup \left\{ A \in V \;\middle|\; \begin{array}{l} A \to BC \in P, \\ B \in V(i,k), \\ C \in V(i+k, j-k) \end{array} \right\}$
    wenn $S \in V(1,n)$ dann
        return Ja
    sonst
        return Nein
```

Laufzeit des CYK-Algorithmus

- Drei geschachtelte Für-Schleifen
- Im Inneren wird noch über alle Produktionen aus P iteriert
- Mit $n = |w|$ und $|P| =$ Anzahl der Iterationen kann die Laufzeitkomplexität mit $\mathcal{O}(n^3 \cdot |P|)$ abgeschätzt werden.

Theorem

Das Wortproblem für kontextfreie Sprachen kann in Polynomialzeit entschieden werden.

Beispiel

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$
$V(i,j)$-Tabelle ist zunächst leer:

i →

	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1					
2					
3					
4					
5					

j ↓

Beispiel (2)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$
Füllen der $V(i, 1)$-Einträge:

i →

j ↓

	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1					
2					
3					
4					
5					

Beispiel (2)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$
Füllen der $V(i, 1)$-Einträge:

b b d d c

$i \longrightarrow$

$j \downarrow$

$V(i,j)$	1	2	3	4	5
1	B				
2					
3					
4					
5					

Beispiel (2)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$
Füllen der $V(i, 1)$-Einträge:

b b d d c

$i \longrightarrow$

$j \downarrow$

$V(i,j)$	1	2	3	4	5
1	B	B			
2					
3					
4					
5					

Beispiel (2)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$
Füllen der $V(i, 1)$-Einträge:

	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1	B	B	D		
2					
3					
4					
5					

i

j

Beispiel (2)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$
Füllen der $V(i, 1)$-Einträge:

b b d d c

i

j

$V(i,j)$	1	2	3	4	5
1	B	B	D	D	
2					
3					
4					
5					

Beispiel (2)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$
Füllen der $V(i, 1)$-Einträge:

$i \longrightarrow$

	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2					
3					
4					
5					

$j \downarrow$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

$i \longrightarrow$

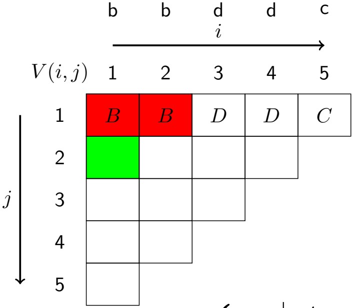
	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2					
3					
4					
5					

$j \downarrow$

3-fach Schleife:

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

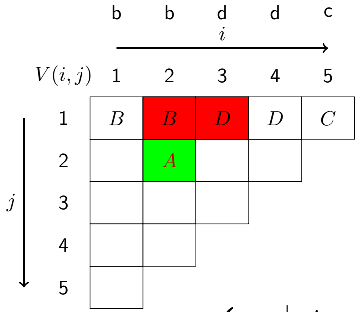


$V(i,j)$	1	2	3	4	5
1	🟥 B	🟥 B	D	D	C
2	🟩				
3					
4					
5					

3-fach Schleife:j=2,i=1,k=1: $V(1,2) = V(1,2) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(1,1), \\ C \in V(2,1) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

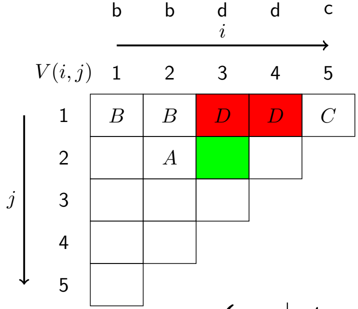


$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2		A			
3					
4					
5					

3-fach Schleife:j=2,i=2,k=1: $V(2,2) = V(2,2) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(2,1), \\ C \in V(3,1) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

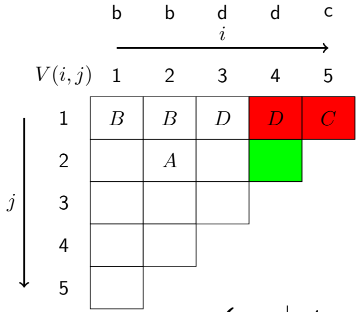


$V(i,j)$	1	2	3	4	5
1	B	B	🟥 D	🟥 D	C
2		A	🟩		
3					
4					
5					

3-fach Schleife:j=2,i=3,k=1: $V(3,2) = V(3,2) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(3,1), \\ C \in V(4,1) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

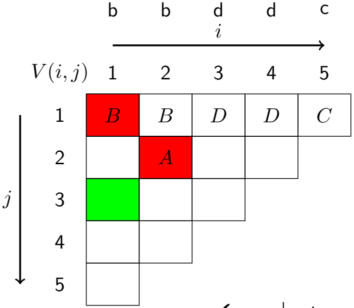


$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2		A		🟩	
3					
4					
5					

3-fach Schleife:j=2,i=4,k=1: $V(4,2) = V(4,2) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(4,1), \\ C \in V(5,1) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

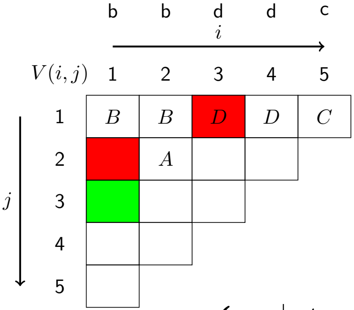


	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2		A			
3	🟩				
4					
5					

3-fach Schleife:j=3,i=1,k=1: $V(1,3) = V(1,3) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(1,1), \\ C \in V(2,2) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

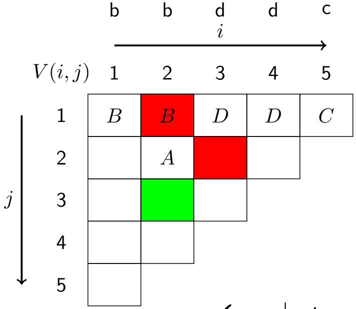


$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2	🟥	A			
3	🟩				
4					
5					

3-fach Schleife:j=3,i=1,k=2: $V(1,3) = V(1,3) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(1,2), \\ C \in V(3,1) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S,A,B\},\{b,c,d\},P,S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

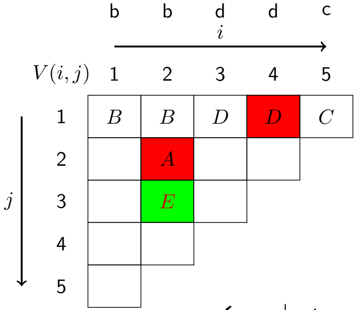


	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2		A	🟥		
3		🟩			
4					
5					

3-fach Schleife:j=3,i=2,k=1: $V(2,3) = V(2,3) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(2,1), \\ C \in V(3,2) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

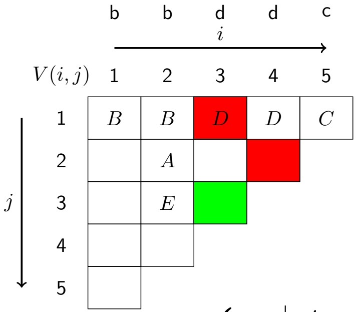


	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2		A			
3		E			
4					
5					

3-fach Schleife:j=3,i=2,k=2: $V(2,3) = V(2,3) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(2,2), \\ C \in V(4,1) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

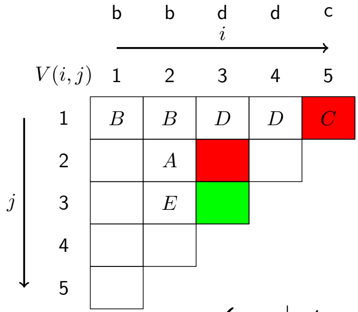


$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2		A		🟥	
3		E	🟩		
4					
5					

3-fach Schleife:j=3,i=3,k=1: $V(3,3) = V(3,3) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(3,1), \\ C \in V(4,2) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

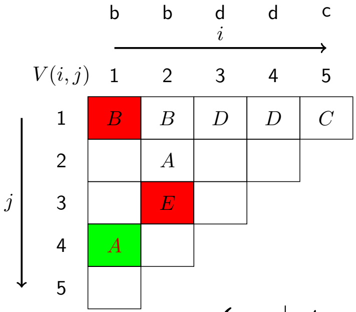


	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2		A	🟥		
3		E	🟩		
4					
5					

3-fach Schleife:j=3,i=3,k=2: $V(3,3) = V(3,3) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(3,2), \\ C \in V(5,1) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ \color{red}{A \to BE},\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

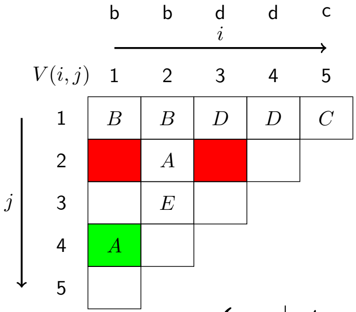


$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2		A			
3		E			
4	A				
5					

3-fach Schleife:j=4,i=1,k=1: $V(1,4) = V(1,4) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(1,1), \\ C \in V(2,3) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

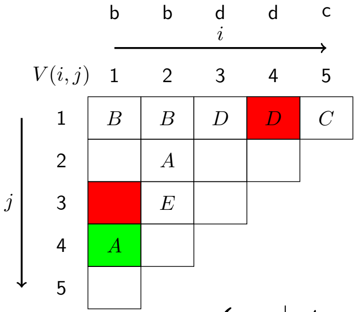


$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2	🟥	A	🟥		
3		E			
4	🟩 A				
5					

3-fach Schleife:j=4,i=1,k=2: $V(1,4) = V(1,4) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(1,2), \\ C \in V(3,2) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

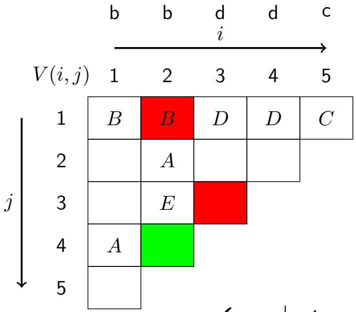


		b	b	d	d	c
	$V(i,j)$	1	2	3	4	5
	1	B	B	D	D	C
	2		A			
j	3	🟥	E			
	4	A				
	5					

3-fach Schleife:j=4,i=1,k=3: $V(1,4) = V(1,4) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(1,3), \\ C \in V(4,1) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

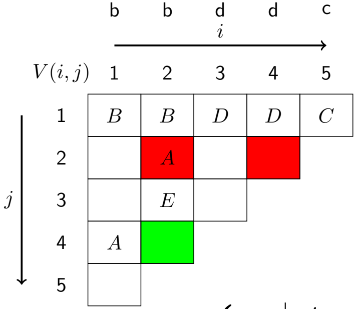


	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2		A			
3		E	🟥		
4	A	🟩			
5					

3-fach Schleife:j=4,i=2,k=1: $V(2,4) = V(2,4) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(2,1), \\ C \in V(3,3) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

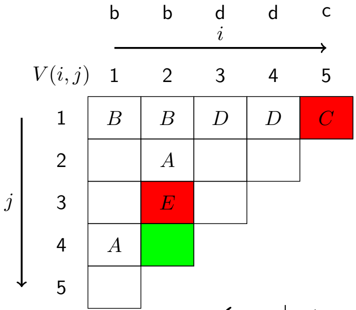


	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2		🟥 A		🟥	
3		E			
4	A	🟩			
5					

3-fach Schleife:j=4,i=2,k=2: $V(2,4) = V(2,4) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(2,2), \\ C \in V(4,2) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

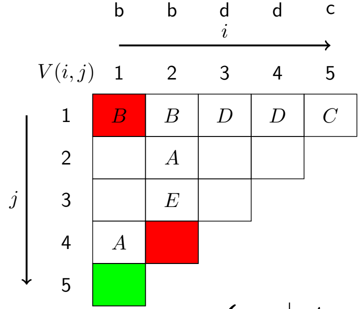


	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2		A			
3		E			
4	A	🟩			
5					

3-fach Schleife:j=4,i=2,k=3: $V(2,4) = V(2,4) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(2,3), \\ C \in V(5,1) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

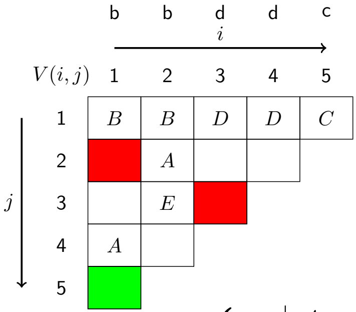


	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1	🟥 B	B	D	D	C
2		A			
3		E			
4	A	🟥			
5	🟩				

3-fach Schleife:j=5,i=1,k=1: $V(1,5) = V(1,5) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(1,1), \\ C \in V(2,4) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

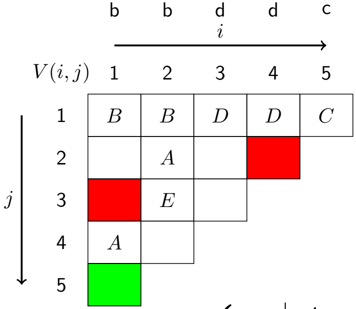


	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1	B	B	D	D	C
2	🟥	A			
3		E	🟥		
4	A				
5	🟩				

3-fach Schleife:j=5,i=1,k=2: $V(1,5) = V(1,5) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(1,2), \\ C \in V(3,3) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

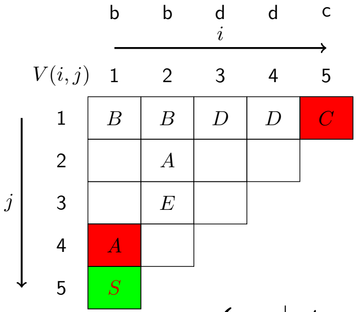


		b	b	d	d	c
	$V(i,j)$	1	2	3	4	5
	1	B	B	D	D	C
	2		A		🟥	
j	3	🟥	E			
	4	A				
	5	🟩				

3-fach Schleife:j=5,i=1,k=3: $V(1,5) = V(1,5) \cup \left\{ A \,\middle|\, \begin{array}{l} A \to BC, \\ B \in V(1,3), \\ C \in V(4,2) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$



	b	b	d	d	c
$V(i,j)$	1	2	3	4	5
1	B	B	D	D	🟥 C
2		A			
3		E			
4	🟥 A				
5	🟩 S				

3-fach Schleife:j=5,i=1,k=4: $V(1,5) = V(1,5) \cup \left\{ A \;\middle|\; \begin{array}{l} A \to BC, \\ B \in V(1,4), \\ C \in V(5,1) \end{array} \right\}$

Beispiel (3)

Sei $w = bbddc$ und $G = (\{S, A, B\}, \{b, c, d\}, P, S)$ mit
$P = \{S \to AC,\ A \to BE,\ A \to BD,\ E \to AD,\ C \to c,\ B \to b,\ D \to d\}$

b b d d c

i →

j ↓	$V(i,j)$	1	2	3	4	5
	1	B	B	D	D	C
	2		A			
	3		E			
	4	A				
	5	S				

Da $S \in V(1,5)$ gilt $w \in L(G)$

Web-Tool zum Üben / Anschauen

`http://www.cip.ifi.lmu.de/~lindebar/`

Quiz 3

- Besuchen Sie `http://www.cip.ifi.lmu.de/~lindebar/`
- Wählen Sie 3V pro T als Beispiel
- Erzeugen Sie die CYK-Tabelle für aabaab

Entscheiden Sie anhand der Tabelle: Welche Wörter werden von G erzeugt?

- aabaab
- aabaa
- baab
- abaa
- aba

arsnova.hs-rm.de

6750 1376