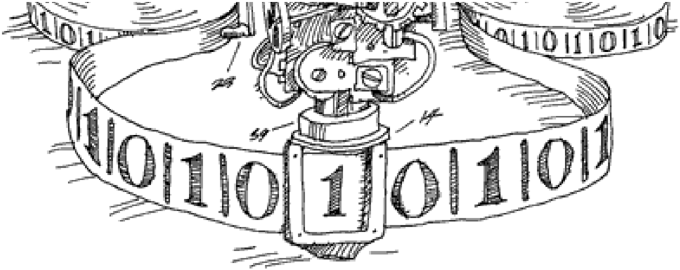
Karlsruher Institut für Technologie

Theoretische Grundlagen der Informatik

Tutorium 3

Institut für Kryptographie und Sicherheit



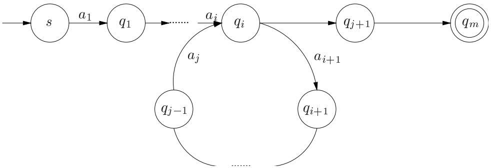
Pumping Lemma

Sei L eine reguläre Sprache. Dann existiert eine Zahl $p \in \mathbb{N}$, sodass für jedes Wort $w \in L$ mit $|w| > p$ eine Darstellung

$$w = xyz$$

existiert, so dass folgende Eigenschaften erfüllt sind:

1. $y \neq \epsilon$
2. $|xy| \leq p$
3. Für alle $i \in \mathbb{N}_0$ gilt: $xy^iz \in L$



Pumping Lemma Formalia

Behauptung: $L = \{\_\_\_\_\_\_\_\_\}$ ist nicht regulär.
Beweis: Annahme L sei regulär:

Sei $p \in \mathbb{N}$ wie im Pumping-Lemma

Wähle w = ________, $w \in L$, $|w| > p$

Beh: $\forall x, y, z : w = xyz, |xy| \leq p, y \neq \epsilon$ gilt: $\exists i \in \mathbb{N}_0 : xy^iz \notin L$

Bew: $(\forall y$ gilt:$)$________________________

____

Widerspruch zum Pumping Lemma $\Rightarrow$ L ist nicht regulär.

Aufgabe 1

Gegeben sei die Sprache
$\mathcal{L} = \{w \in \{a, b\}^* \mid w \text{ enthält gleich viele } a \text{ wie } b\}$.

1. Wie lautet das Pumping Lemma? Was genau muss man zeigen, falls man die Kontraposition des Pumping Lemmas verwenden will?
2. Zeigen Sie mit Hilfe des Pumping Lemmas, dass L nicht regulär ist!
3. Zeigen Sie mit Hilfe des Pumping Lemmas, dass die Sprache $\mathcal{L}' = \{a^p \mid p \text{ Primzahl}\}$ nicht regulär ist.
4. Betrachten Sie nun die Sprache $\mathcal{L}'' = \{a, aab, aaab\}$! Ist diese regulär? Falls ja, geben Sie einen endlichen Automaten an, der diese Sprache akzeptiert! Kann man mit dem Pumping Lemma zeigen, dass die Sprache regulär ist?

Kontextfreie Sprachen

- Echte Obermenge von regulären Sprachen
- Auch Chompsky Typ 2 genannt
- Wird von Kontextfreien Grammatiken $G = (\Sigma, V, S, P)$ gesprochen

Σ	endliches Alphabet der Terminalsymbole
V	endliches Alphabet der Variablen ($V \cap \Sigma = \emptyset$)
S	Startsymbol $\in V$
P	Produktionsmenge, mit Produktionsform $u \to v$, $u \in V$ und $v \in \{\Sigma \cap V\}^*$

- Können von Kellerautomaten erkannt werden

Chomsky-Normalform

Der Cocke-Younger-Kasami-Algorithmus (CYK) löst das Wortproblem für kontextfreie Sprachen (CH-2) in $\mathcal{O}(n^3)$. Um CYK anzuwenden, muss die gegebene Grammatik erst in Chomsky-Normalform gebracht werden. Das ist für jede CH-2-Grammatik möglich.

Chomsky-Normalform

Eine CH-2-Grammatik $G = (\Sigma, \mathcal{V}, \mathcal{S}, \mathcal{R})$ ist in Chomsky-Normalform, wenn jede Produktion aus $\mathcal{R}$ eine der folgenden Formen hat:

- $A \rightarrow BC$
- $A \rightarrow a$

Wobei gilt $A, B, C \in \mathcal{V}$ und $a \in \Sigma$. Um das leere Wort in der Sprache zu erlauben, lässt sich die Grammatik leicht mit neuem Startsymbol S' ergänzen mit der Regel

$$S' \rightarrow S \mid \epsilon$$

Umwandlung in Chomsky-Normalform

1. Für alle $a \in \Sigma$ und für alle Produktionen, auf deren rechter Seite a vorkommt (außer für $V \to a$, mit $V \in \mathcal{V}$), wird jedes Vorkommen von a durch ein *neues* Nichtterminalsymbol A ersetzt und die Produktion $A \to a$ wird hinzugefügt.

Umwandlungsbeispiel (Schritt 1 von 4)

$$\begin{aligned} \mathcal{S} &\to XY \\ X &\to aXb \mid Z \mid \epsilon \\ Y &\to ccY \mid \epsilon \\ Z &\to X \end{aligned} \quad \Rightarrow \quad \begin{aligned} \mathcal{S} &\to XY \\ X &\to AXB \mid Z \mid \epsilon \\ Y &\to CCY \mid \epsilon \\ Z &\to X \\ A &\to a \\ B &\to b \\ C &\to c \end{aligned}$$

Umwandlung in Chomsky-Normalform

2. Für Produktionen mit mehr als zwei Variablen rechts werden *neue* Nichterminale eingeführt und dazu passende Produktionen hinzugefügt.

Umwandlungsbeispiel (Schritt 2 von 4)

$$
\begin{aligned}
\mathcal{S} &\rightarrow XY \\
X &\rightarrow AXB \mid Z \mid \epsilon \\
Y &\rightarrow CCY \mid \epsilon \\
Z &\rightarrow X \\
A &\rightarrow a \\
B &\rightarrow b \\
C &\rightarrow c
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
\mathcal{S} &\rightarrow XY \\
X &\rightarrow FB \mid Z \mid \epsilon \\
Y &\rightarrow GY \mid \epsilon \\
Z &\rightarrow X \\
F &\rightarrow AX \\
G &\rightarrow CC \\
A &\rightarrow a \\
B &\rightarrow b \\
C &\rightarrow c
\end{aligned}
$$

Umwandlung in Chomsky-Normalform

3. Entfernen von Produktionen der Form $V \to \epsilon$ für $V \in \mathcal{V}, v \neq \mathcal{S}$
$\Rightarrow$ „Vorwegnahme“ dieser Produktionen: Für jede Produktion mit einem der obigen V auf der rechten Seite wird eine neue Produktion ohne dieses V hinzugefügt.

Umwandlungsbeispiel (Schritt 3 von 4)

$$
\begin{aligned}
\mathcal{S} &\to XY \\
X &\to FB \mid Z \mid \epsilon \\
Y &\to GY \mid \epsilon \\
Z &\to X \\
F &\to AX \\
G &\to CC \\
A &\to a, B \to b, C \to c
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
\mathcal{S} &\to XY \mid X \mid Y \mid \epsilon \\
X &\to FB \mid Z \mid \cancel{\epsilon} \\
Y &\to GY \mid G \mid \cancel{\epsilon} \\
Z &\to X \\
F &\to AX \mid A \\
G &\to CC \\
A &\to a, B \to b, C \to c
\end{aligned}
$$

Umwandlung in Chomsky-Normalform

Karlsruher Institut für Technologie

4. Für Produktionen mit einer Variablen rechts werden Zyklen gesucht, für gefundene Zyklen werden alle Vorkommnisse aller Variablen des Zyklus durch einen Repräsentanten ausgetauscht. Danach werden triviale Produktionen entfernt.

Umwandlungsbeispiel (Schritt 4a von 4)

$$
\begin{array}{l}
\mathcal{S} \to XY \mid X \mid Y \mid \epsilon \\
X \to FB \mid Z \\
Y \to GY \mid G \\
Z \to X \\
F \to AX \mid A \\
G \to CC \\
A \to a, B \to b, C \to c
\end{array}
\quad \Rightarrow \quad
\begin{array}{l}
\mathcal{S} \to XY \mid X \mid Y \mid \epsilon \\
X \to FB \mid \cancel{X} \\
Y \to GY \mid G \\
\cancel{X} \to \cancel{X} \\
F \to AX \\
G \to CC \\
A \to a, B \to b, C \to c
\end{array}
$$

Umwandlung in Chomsky-Normalform

4. Alle Regeln, die rechts eine einzelne Variable haben, werden durch „Vorziehen“ der Regeln eliminiert.
   Außerdem wird ein neues Startsymbol eingeführt, falls eine Regel $\mathcal{S} \rightarrow \epsilon$ existiert.

Umwandlungsbeispiel (Schritt 4b von 4)

$$
\begin{aligned}
\mathcal{S} &\rightarrow XY \mid X \mid Y \mid \epsilon \\
X &\rightarrow FB \\
Y &\rightarrow GY \mid G \\
F &\rightarrow AX \\
G &\rightarrow CC \\
A &\rightarrow a \\
B &\rightarrow b \\
C &\rightarrow c
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
\mathcal{S}' &\rightarrow \mathcal{S} \mid \epsilon \\
\mathcal{S} &\rightarrow XY \mid FB \mid GY \mid CC \\
X &\rightarrow FB \\
Y &\rightarrow GY \mid CC \\
F &\rightarrow AX \\
G &\rightarrow CC \\
A &\rightarrow a \\
B &\rightarrow b \\
C &\rightarrow c
\end{aligned}
$$

Chomsky-Normalform

1. Produktionen auf Terminale und Nicht-Terminale sortieren
2. Produktionen auf mehr als zwei Nicht-Terminale ersetzen
3. Produktionen auf ϵ ersetzen
4. Produktionen auf ein Nicht-Terminal ersetzen
5. Neues Startsymbol einführen falls $S \rightarrow \epsilon$ existiert
6. ???
7. PROFIT

Aufgabe 2

Gegeben sei die folgende Grammatik: $\mathcal{G} = (\mathcal{T}, \mathcal{V}, S, \mathcal{P})$ mit
$\mathcal{T} := \{a, b, c, d\}$, $\mathcal{V} := \{S, A, D, M\}$,
$\mathcal{P} := \{S \to AMD \mid M, A \to AA \mid a, D \to DD \mid d, M \to bMc \mid \epsilon\}$

1. Geben Sie die erzeugte Sprache an!
2. Wandeln Sie die gegebene kontextfreie Grammatik $\mathcal{G}$ in eine äquivalente kontextfreie Grammatik $\mathcal{G}'$ in Chomsky-Normalform um, indem sie jeden Schritt durch eine neue Grammatik beschreiben!

CYK Überblick

CYK ist ein Algorithmus, um das Wortproblem in CH-2 zu lösen. Um den Algorithmus anzuwenden, muss eine Grammatik in Chomsky-Normalform vorliegen.
Grundidee zur Überprüfung eines Wortes der Länge n:

- Wir betrachen $V_{i,j} =$ Menge der Nichtterminale, aus denen das Teilwort der Position i bis j abgeleitet werden kann
- Die Frage, ob $V_{i,j}$ ableitbar ist, lässt sich entscheiden durch Betrachten aller möglichen $V_{i,k}$ und $V_{k+1,j}$
- $V_{i,i}$ sind trivial
- Bottom-up lässt sich dadurch $V_{1,n}$ berechnen
- Ist $S \in V_{1,n}$, so lässt sich das Wort ableiten

CYK Beispiel

Gegeben sei die Grammatik $G = (\mathcal{T}, \mathcal{V}, S, \mathcal{P})$ mit den folgenden Produktionen aus $\mathcal{P}$:

$$S \to AX \mid AB$$
$$X \to SB \mid AB$$
$$A \to a$$
$$B \to b$$

1. Lässt sich der CYK-Algorithmus auf G anwenden?
2. Ist das Wort *aaabbb* in der Sprache $\mathcal{L}(G)$?

Aufgabe 2 Fortsetzung

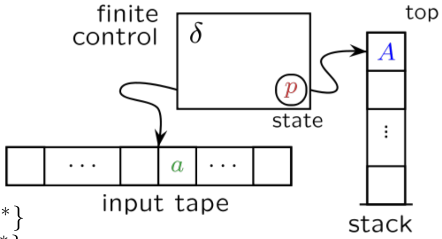
Gegeben sei die folgende Grammatik: $\mathcal{G} = (\mathcal{T}, \mathcal{V}, S, \mathcal{P})$ mit
$\mathcal{T} := \{a, b, c, d\}$, $\mathcal{V} := \{S, A, D, M\}$,
$\mathcal{P} := \{S \to AMD \mid M, A \to AA \mid a, D \to DD \mid d, M \to bMc \mid \epsilon\}$

1. Geben Sie die erzeugte Sprache an!
2. Wandeln Sie die gegebene kontextfreie Grammatik $\mathcal{G}$ in eine äquivalente kontextfreie Grammatik $\mathcal{G}'$ in Chomsky-Normalform um, indem sie jeden Schritt durch eine neue Grammatik beschreiben!
3. Zeigen oder widerlegen Sie mit Hilfe des CYK-Algorithmus, ob die folgenden Wörter in der Sprache $\mathcal{L}$ liegen, die durch die Grammatik $\mathcal{G}$ erzeugt wird!
   3.1 *aabbccdd*
   3.2 *abbcc*
   3.3 *abcdd*

Definition Kellerautomaten

Ein (nichtdeterministischer) **Kellerautomat** (NPDA bzw PDA, Pushdown Automaton) besteht aus $(Q, \Sigma, \Gamma, q_0, Z_0, \delta, F)$, wobei

- Q endliche Zustandsmenge
- Σ endliches Eingabealphabet
- Γ endliches Stack-Alphabet
- $q_0 \in Q$ Anfangszustand
- $Z_0 \in \Gamma$ Initialisierung des Stacks
- $\delta : Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \to 2^{Q \times \Gamma^*}$
  - $\delta(q, a, Z) \subseteq \{(q, \gamma) : q \in Q, \gamma \in \Gamma^*\}$
  - $\delta(q, \epsilon, Z) \subseteq \{(q, \gamma) : q \in Q, \gamma \in \Gamma^*\}$
- $F \subseteq Q$ Menge der akzeptierenden Endzustände, $F = \emptyset$ ist möglich.



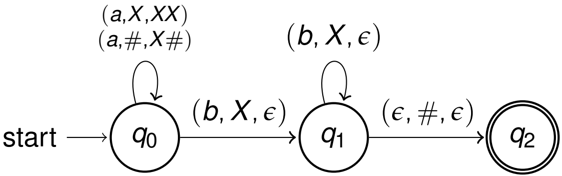
Zu Kellerautomaten

- Akzeptieren nach Eingabeende, wenn
  - der Stack leer ist *oder*
  - der Automat in einen akzeptierenden Zustand kommt.
- Sind im Allgemeinen nichtdeterministisch
- Man kann Endzustände auch aus der Definition weglassen und alternativ verlangen, dass der Automat genau bei leerem Keller akzeptiert.
- Man kann sogar alle Zustände bis auf einen weglassen und alles in die Kellerbelegung kodieren

Beispiel

$M = (Q, \Sigma, \Gamma, q_0, Z_0, \delta, F)$

- $Q = \{q_0, q_1, q_2\}$
- $\Sigma = \{a, b\}$
- $\Gamma = \{\#, X\}$
- $Z_0 = \#$
- $F = \{q_2\}$



- Welche Sprache akzeptiert dieser Automat?

Aufgabe 3

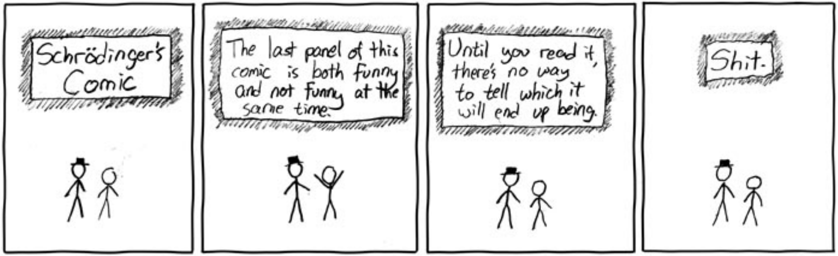
Gegeben sei folgende Sprache für das Alphabet $\Sigma = \{a, b, c\}$:

$$\mathcal{L} = \{w_1 w_2 \in \Sigma^* \mid w_1 \in \{a, b\}^*, w_2 \in \{b, c\}^*, \\ \#_a w_1 + \#_b w_1 = \#_b w_2 + \#_c w_2\}$$

Hier gibt $\#_x w$ die Häufigkeit des Vorkommens eines Zeichens $x \in \Sigma$ in einem Wort $w \in \Sigma^*$ an.

1. Zeigen Sie, dass $\mathcal{L}$ nicht regulär ist!
2. Geben Sie eine Chomsky-2-Grammatik an, die genau die Sprache $\mathcal{L}$ erzeugt!
3. Geben Sie einen Kellerautomaten $\mathcal{M}$ an, der genau die Sprache $\mathcal{L}$ erkennt! Zeichnen Sie den Zustandsübergangsgraphen für $\mathcal{M}$!

Bis zum nächsten Mal!



Lizenzen