3.10 Entscheidbarkeit

Beispiel 55

Wie wir bereits wissen, ist das Wortproblem für reguläre Sprachen L entscheidbar. Wenn L durch einen deterministischen endlichen Automaten gegeben ist, ist dies sogar in linearer Laufzeit möglich. Allerdings gilt, dass bei der Überführung eines nichtdeterministischen endlichen Automaten in einen deterministischen endlichen Automaten die Komplexität exponentiell zunehmen kann.

Die folgenden Probleme sind für Chomsky-3-Sprachen (also die Familie der regulären Sprachen) entscheidbar:

Wortproblem: Ist ein Wort w in $L(G)$ (bzw. $L(A)$)?

Das Wortproblem ist für alle Sprachen mit einem Chomsky-Typ größer 0 entscheidbar. Allerdings wächst die Laufzeit exponentiell mit der Wortlänge n. Für Chomsky-2- und Chomsky-3-Sprachen gibt es wesentlich effizientere Algorithmen.

Leerheitsproblem: Ist $L(G) = \emptyset$?

Das Leerheitsproblem ist für Sprachen vom Chomsky-Typ 2 und 3 entscheidbar.
Für andere Sprachtypen lassen sich Grammatiken konstruieren, für die nicht mehr entscheidbar ist, ob die Sprache leer ist.

Endlichkeitsproblem: Ist $|L(G)| < \infty$?

Das Endlichkeitsproblem ist für alle regulären Sprachen lösbar.

Lemma 56

Sei n die Pumping-Lemma-Zahl, die zur regulären Sprache L gehört. Dann gilt:

$$|L| = \infty \text{ gdw } (\exists z \in L)[n \leq |z| < 2n] \; .$$

Beweis:

Wir zeigen zunächst $\Leftarrow$:
Aus dem Pumping-Lemma folgt: $z = uvw$ für $|z| \geq n$ und $uv^iw \in L$ für alle $i \in \mathbb{N}_0$. Damit erzeugt man unendlich viele Wörter.

Nun wird $\Rightarrow$ gezeigt:
Dass es ein Wort z mit $|z| \geq n$ gibt, ist klar (es gibt ja unendlich viele Wörter). Mit Hilfe des Pumping-Lemmas lässt sich ein solches Wort auf eine Länge $< 2n$ reduzieren. $\square$

Damit kann das Endlichkeitsproblem auf das Wortproblem zurückgeführt werden.

Schnittproblem: Ist $L(G_1) \cap L(G_2) = \emptyset$?

Das Schnittproblem ist für die Familie der regulären Sprachen entscheidbar, nicht aber für die Familie der Chomsky-2-Sprachen.

Äquivalenzproblem: Ist $L(G_1) = L(G_2)$?

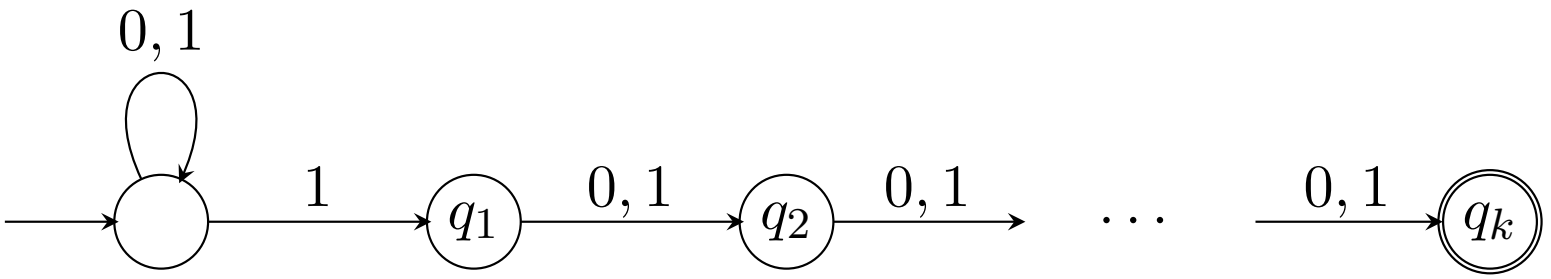
Das Äquivalenzproblem lässt sich auch wie folgt formulieren:

$$L_1 = L_2 \quad \Leftrightarrow \quad (L_1 \cap \overline{L_2}) \cup (L_2 \cap \overline{L_1}) = \emptyset$$

Wichtig für eine effiziente Lösung der Probleme ist, wie die Sprache gegeben ist. Hierzu ein Beispiel:

Beispiel 57

$L = \{w \in \{0,1\}^*$; das k-letzte Bit von w ist gleich $1\}$
Ein NFA für diese Sprache ist gegeben durch:



Insgesamt hat der NFA $k+1$ Zustände. Man kann nun diesen NFA in einen deterministischen Automaten umwandeln und stellt fest, dass der entsprechende DFA $\Omega(2^k)$ Zustände hat.

Da die Komplexität eines Algorithmus von der Größe der Eingabe abhängt, ist dieser Unterschied in der Eingabegröße natürlich wesentlich, denn es gilt:

kurze Eingabe wie beim NFA $\Rightarrow$ wenig Zeit für einen effizienten Algorithmus,

lange Eingabe wie beim DFA $\Rightarrow$ mehr Zeit für einen effizienten Algorithmus.

4. Kontextfeie Grammatiken und Sprachen

4.1 Grundlagen und ein Beispiel

Sei

$$L_= := \{w \in \{0,1\}^*;\ w \text{ enthält gleich viele 0en und 1en}\}.$$

Sei $\#_a(w)$ die Anzahl der Zeichen a in der Zeichenreihe w, d.h.

$$L_= = \{w \in \{0,1\}^*;\ \#_0(w) = \#_1(w)\}.$$

$L_=$ ist sicherlich nicht regulär (vgl. Pumping-Lemma).

Satz 58

Die (kontextfreie) Grammatik G

$$S \to \epsilon \mid T$$
$$T \to TT \mid 0T1 \mid 1T0 \mid 01 \mid 10$$

erzeugt $L_=$.

Beweis:

Sei $w \in L_=$. Betrachte für jedes Präfix x von w die Zahl

$$\#_1(x) - \#_0(x) \ .$$

Falls $w = w'w''$ für nichtleere $w', w'' \in L_=$, wende man Induktion über $|w|$ an, falls nicht, ist w von der Form $0w'1$ oder $1w'0$, und Induktion liefert wiederum die Behauptung. □

Definition (Wiederholung, siehe Def. 19)

- *Eine kontextfreie Grammatik G heißt eindeutig, wenn es für jedes $w \in L(G)$ genau einen Ableitungsbaum gibt.*
- *Eine kontextfreie Sprache L heißt eindeutig, falls es eine eindeutige kontextfreie Grammatik G mit $L = L(G)$ gibt. Ansonsten heißt L inhärent mehrdeutig.*

Die oben angegebene Grammatik für $L_=$ ist nicht eindeutig.

4.2 Die Chomsky-Normalform

Sei $G = (V, \Sigma, P, S)$ eine kontexfreie Grammatik.

Definition 59

Eine kontextfreie Grammatik G ist in Chomsky-Normalform, falls alle Produktionen eine der Formen

$$\begin{aligned} A &\to a & A &\in V, a \in \Sigma, \\ A &\to BC & A, B, C &\in V, \text{ oder} \\ S &\to \epsilon \end{aligned}$$

haben.

Algorithmus zur Konstruktion einer (äquivalenten) Grammatik in Chomsky-Normalform

Eingabe: Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$

1. Wir fügen für jedes $a \in \Sigma$ zu V ein neues Nichtterminal Y_a hinzu, ersetzen in allen Produktionen a durch Y_a und fügen $Y_a \to a$ als neue Produktion zu P hinzu.

   /* linearer Zeitaufwand, Größe vervierfacht sich höchstens */

2. Wir ersetzen jede Produktion der Form

   $$A \to B_1 B_2 \cdots B_r \ (r \geq 3)$$

   durch

   $$A \to B_1 C_2,\ C_2 \to B_2 C_3, \ldots, C_{r-1} \to B_{r-1} B_r,$$

   wobei $C_2, \ldots, C_{r-1}$ neue Nichtterminale sind.

   /* linearer Zeitaufwand, Größe vervierfacht sich höchstens */

3. Für alle $C, D \in V$, $C \neq D$, mit

$$C \to^+ D,$$

füge für jede Produktion der Form

$$A \to BC \in P \text{ bzw. } A \to CB \in P$$

die Produktion

$$A \to BD \text{ bzw. } A \to DB$$

zu P hinzu.

/* quadratischer Aufwand pro A */

4. Für alle $\alpha \in L(G) \cap (V^2 \cup \Sigma)$, füge $S \to \alpha$ zu P hinzu.
5. Streiche alle Produktionen der Form $A \to B$ aus P.

Zusammenfassend können wir festhalten:

Satz 60

Aus einer kontextfreien Grammatik $G = (V, \Sigma, P, S)$ der Größe $s(G)$ kann in Zeit $O(|V|^2 \cdot s(G))$ eine äquivalente kontextfreie Grammatik in Chomsky-Normalform der Größe $O(|V|^2 \cdot s(G))$ erzeugt werden.

4.3 Der Cocke-Kasami-Younger-Algorithmus

Der CYK-Algorithmus entscheidet das Wortproblem für kontextfreie Sprachen, falls die Sprache in Form einer Grammatik in Chomsky-Normalform gegeben ist.

Eingabe: Grammatik $G = (V, \Sigma, P, S)$ in Chomsky-Normalform, $w = w_1 \dots w_n \in \Sigma^*$ mit der Länge n. O.B.d.A. $n > 0$.

Definition

$$V_{ij} := \{A \in V;\ A \to^* w_i \dots w_j\} .$$

Es ist klar, dass $w \in L(G) \Leftrightarrow S \in V_{1n}$.

Der CYK-Algorithmus berechnet alle V_{ij} rekursiv nach wachsendem $j - i$. Den Anfang machen die

$$V_{ii} := \{A \in V;\ A \to w_i \in P\},$$

der rekursive Aufbau erfolgt nach der Regel

$$V_{ij} = \bigcup_{i \le k < j} \{A \in V;\ (A \to BC) \in P \wedge B \in V_{ik} \wedge C \in V_{k+1,j}\} \quad \text{für } i < j.$$

Die Korrektheit dieses Aufbaus ist klar, wenn die Grammatik in Chomsky-Normalform vorliegt.

Zur Komplexität des CYK-Algorithmus

Es werden $\frac{n^2+n}{2}$ Mengen V_{ij} berechnet. Für jede dieser Mengen werden $|P|$ Produktionen und höchstens n Werte für k betrachtet. Der Test der Bedingung $(A \to BC) \in P \wedge B \in V_{ik} \wedge C \in V_{k+1,j}$ erfordert bei geeigneter Repräsentation der Mengen V_{ij} konstanten Aufwand. Der Gesamtaufwand ist also $O(|P|n^3)$.

Mit der gleichen Methode und dem gleichen Rechenaufwand kann man zu dem getesteten Wort, falls es in der Sprache ist, auch gleich einen Ableitungsbaum konstruieren, indem man sich bei der Konstruktion der V_{ij} nicht nur merkt, welche Nichtterminale sie enthalten, sondern auch gleich, warum sie sie enthalten, d.h. aufgrund welcher Produktionen sie in die Menge aufgenommen wurden.

4.4 Das Pumping-Lemma und Ogdens Lemma für kontextfreie Sprachen

Zur Erinnerung: Das Pumping-Lemma für reguläre Sprachen: Für jede reguläre Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ zerlegen lässt in $z = uvw$ mit $|uv| \leq n$, $|v| \geq 1$ und $uv^*w \subseteq L$.

Zum Beweis haben wir $n = |Q|$ gewählt, wobei Q die Zustandsmenge eines L erkennenden DFA war. Das Argument war dann, dass beim Erkennen von z (mindestens) ein Zustand zweimal besucht werden muss und damit der dazwischen liegende Weg im Automaten beliebig oft wiederholt werden kann.

Völlig gleichwertig kann man argumentieren, dass bei der Ableitung von z mittels einer rechtslinearen Grammatik ein Nichtterminalsymbol (mindestens) zweimal auftreten muss und die dazwischen liegende Teilableitung beliebig oft wiederholt werden kann.

Genau dieses Argument kann in ähnlicher Form auch auf kontextfreie Grammatiken (in Chomsky-Normalform) angewendet werden:

Satz 61 (Pumping-Lemma)

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ zerlegen lässt in

$$z = uvwxy,$$

mit

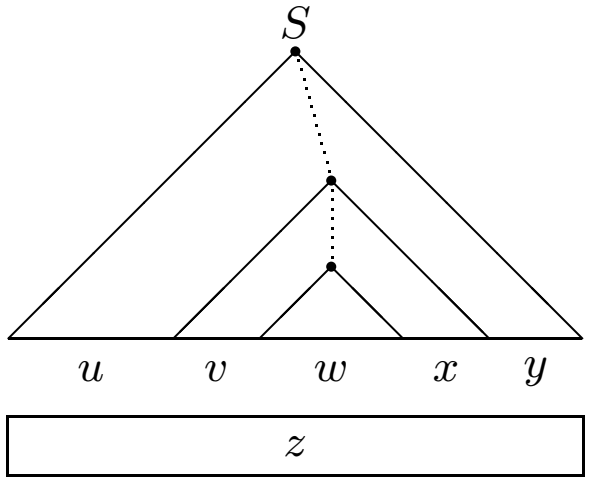
1. $|vx| \geq 1$,
2. $|vwx| \leq n$, *und*
3. $\forall i \in \mathbb{N}_0 : \; uv^i wx^i y \in L$.

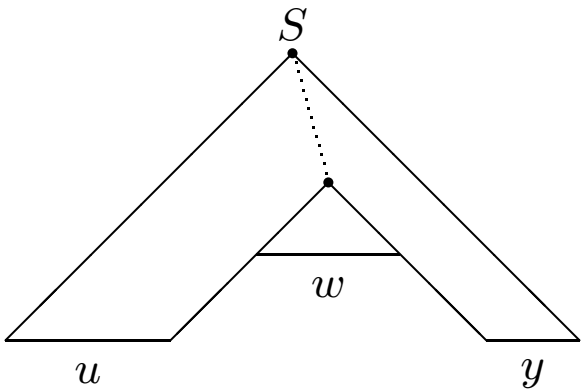
Beweis:

Sei $G = (V, \Sigma, P, S)$ eine Grammatik in Chomsky-Normalform mit $L(G) = L$. Wähle $n = 2^{|V|}$. Sei $z \in L(G)$ mit $|z| \geq n$. Dann hat der Ableitungsbaum für z (ohne die letzte Stufe für die Terminale) mindestens die Tiefe $|V| + 1$, da er wegen der Chomsky-Normalform den Verzweigungsgrad 2 hat.

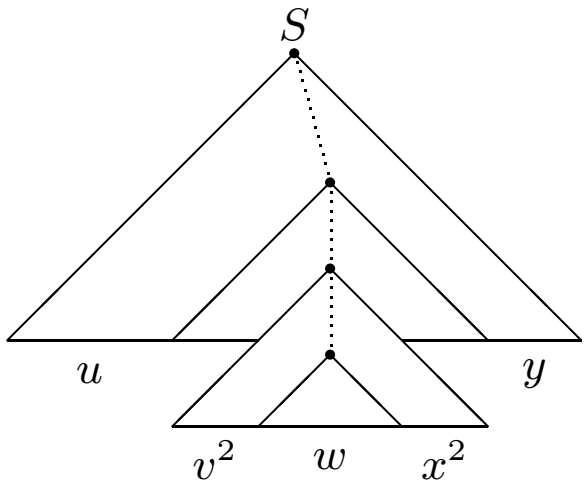
Auf einem Pfadabschnitt der Länge $\geq |V| + 1$ kommt nun mindestens ein Nichtterminal wiederholt vor. Die zwischen diesen beiden Vorkommen liegende Teilableitung kann nun beliebig oft wiederholt werden.

Beweis:





Dieser Ableitungsbaum zeigt
$uwy \in L$



Dieser Ableitungsbaum zeigt
$uv^2wx^2y \in L$

Beweis:

Sei $G = (V, \Sigma, P, S)$ eine Grammatik in Chomsky-Normalform mit $L(G) = L$. Wähle $n = 2^{|V|}$. Sei $z \in L(G)$ mit $|z| \geq n$. Dann hat der Ableitungsbaum für z (ohne die letzte Stufe für die Terminale) mindestens die Tiefe $|V| + 1$, da er wegen der Chomsky-Normalform den Verzweigungsgrad 2 hat.

Auf einem Pfadabschnitt der Länge $\geq |V| + 1$ kommt nun mindestens ein Nichtterminal wiederholt vor. Die zwischen diesen beiden Vorkommen liegende Teilableitung kann nun beliebig oft wiederholt werden.

Um $|vwx| \leq n$ zu erreichen, muss man das am weitesten unten liegende Doppelvorkommen eines solchen Nichtterminals wählen.

□