3.9 Konstruktion minimaler endlicher Automaten

Satz 58

Der nach dem Satz von Myhill-Nerode konstruierte deterministische endliche Automat hat unter allen DFA's für L eine minimale Anzahl von Zuständen.

Beweis:

Sei $A = (Q, \Sigma, \delta, q_0, F)$ mit $L(A) = L$. Dann liefert

$$x \equiv_A y :\Leftrightarrow \hat{\delta}(q_0, x) = \hat{\delta}(q_0, y)$$

eine Äquivalenzrelation, die $\equiv_L$ verfeinert.
Also gilt: $|Q| = \text{index}(\equiv_A) \geq \text{index}(\equiv_L) =$ Anzahl der Zustände des Myhill-Nerode-Automaten. $\square$

Algorithmus zur Konstruktion eines minimalen FA

Eingabe: $A(Q, \Sigma, \delta, q_0, F)$ DFA $\quad (L = L(A))$

Ausgabe: Äquivalenzrelation auf Q.

0. Entferne aus Q alle überflüssigen, d.h. alle von q_0 aus nicht erreichbaren Zustände. Wir nehmen nun an, dass Q keine überflüssigen Zustände mehr enthält.
1. Markiere alle Paare $\{q_i, q_j\} \in Q^2$ mit

$$q_i \in F \text{ und } q_j \notin F \text{ bzw. } q_i \notin F \text{ und } q_j \in F\ .$$

2. **for** alle unmarkierten Paare $\{q_i, q_j\} \in Q^2$, $q_i \neq q_j$ **do**
    if $(\exists a \in \Sigma)[\{\delta(q_i, a), \delta(q_j, a)\}$ ist markiert$]$ **then**
        markiere $\{q_i, q_j\}$;
        for alle $\{q, q'\}$ in $\{q_i, q_j\}$'s Liste **do**
            markiere $\{q, q'\}$ und lösche aus Liste;
            ebenso rekursiv alle Paare in der Liste von $\{q, q'\}$ usw.
        od
    else
        for alle $a \in \Sigma$ **do**
            if $\delta(q_i, a) \neq \delta(q_j, a)$ **then**
                trage $\{q_i, q_j\}$ in die Liste von $\{\delta(q_i, a), \delta(q_j, a)\}$ ein
            fi
        od
    fi
   od
3. Ausgabe: q äquivalent zu $q' \Leftrightarrow \{q, q'\}$ *nicht* markiert.

Satz 59

Obiger Algorithmus liefert einen minimalen DFA für $L(A)$.

Beweis:

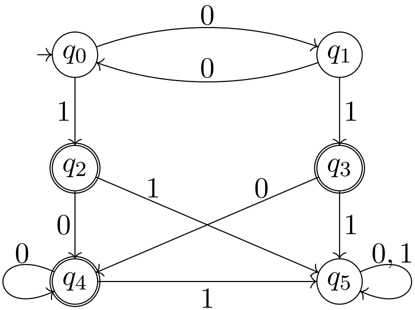
Sei $A' = (Q', \Sigma', \delta', q_0', F')$ der konstruierte Äquivalenzklassenautomat.
Offensichtlich ist $L(A) = L(A')$.
Es gilt: $\{q, q'\}$ wird markiert gdw

$$(\exists\, w \in \Sigma^*)[\hat{\delta}(q, w) \in F \wedge \hat{\delta}(q', w) \notin F \text{ oder umgekehrt}],$$

wie man durch einfache Induktion über $|w|$ sieht.
Also: Die Anzahl der Zustände von A' (nämlich $|Q'|$) ist gleich dem Index von $\equiv_L$. $\square$

Beispiel 60

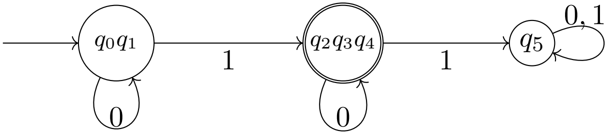
Automat A:



	q_0	q_1	q_2	q_3	q_4	q_5
q_0	/	/	/	/	/	/
q_1		/	/	/	/	/
q_2	×	×	/	/	/	/
q_3	×	×		/	/	/
q_4	×	×			/	/
q_5	×	×	×	×	×	/

Automat A':

$L(A') = 0^*10^*$



Satz 61

Sei $A = (Q, \Sigma, \delta, q_0, F)$ *ein DFA. Der Zeitaufwand des obigen Minimalisierungsalgorithmus ist* $O(|Q|^2|\Sigma|)$.

Beweis:

Für jedes $a \in \Sigma$ muss jede Position in der Tabelle nur konstant oft besucht werden. $\square$

3.10 Entscheidbarkeit

Beispiel 62

Wie wir bereits wissen, ist das Wortproblem für reguläre Grammatiken entscheidbar. Wenn L durch einen deterministischen endlichen Automaten gegeben ist, ist dies (bei festem Alphabet Σ) sogar in linearer Laufzeit möglich. Allerdings gilt, dass die Überführung eines nichtdeterministischen endlichen Automaten in einen deterministischen endlichen Automaten exponentielle Komplexität haben kann.

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

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

Das Wortproblem ist für alle Grammatiken mit einem Chomsky-Typ größer 0 entscheidbar. Allerdings kann die Laufzeit exponentiell mit der Wortlänge n wachsen.

Für Chomsky-2- und Chomsky-3-Sprachen (d.h. -Grammatiken) gibt es wesentlich effizientere Algorithmen.

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

Das Leerheitsproblem ist für Grammatiken vom Chomsky-Typ 2 und 3 entscheidbar.

Für andere Typen 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 Grammatiken lösbar.

Lemma 63

Sei n eine geeignete 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 Klasse der regulären Grammatiken entscheidbar, nicht aber für die Klasse der Chomsky-2-Grammatiken.

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

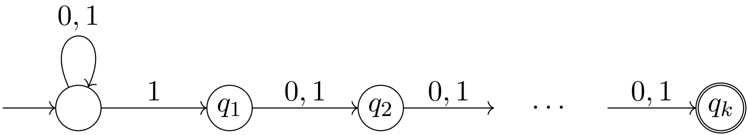
Das Äquivalenzproblem lässt sich auch wie folgt formulieren:

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

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

Beispiel 64

$L = \{w \in \{0,1\}^*;$ 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.

Es gilt allerdings, dass sich für das Wortproblem (uniform oder nicht-uniform) kein großer Komplexitätsunterschied in Abhängigkeit davon ergibt, ob die Sprache durch einen NFA oder DFA dargestellt ist.