Automaten und Formale Sprachen Tutorium

Teil II.

Christopher Blöcker (inf8871)

FH Wedel, SS 2010

Automaten

Äquivalenz

Definition

Zwei Automaten $\mathcal{A}$ und $\mathcal{A}'$ sind äquivalent, wenn sie dieselbe Sprache akzeptieren.

$$\mathcal{A} \simeq \mathcal{A}' \Leftrightarrow \mathcal{L}_{\mathcal{A}} = \mathcal{L}_{\mathcal{A}'}.$$

Konstruktion

Ein zu einem **NEA** $\mathcal{A}$ äquivalenter **DEA** $\mathcal{A}'$ kann durch die Potenzmengenkonstruktion bestimmt werden.

Potenzmengenkonstruktion

Sei $\mathcal{A}$ ein **NEA** mit

$$\mathcal{A} = (\mathcal{Q}, \Sigma, \delta, q_0, \mathcal{F}).$$

Der **DEA** $\mathcal{A}'$ mit

$$\mathcal{A}' = (\mathcal{Q}', \Sigma', \delta', q_0', \mathcal{F}') \text{ und } \mathcal{A} \simeq \mathcal{A}'$$

wird bestimmt durch

1. $\mathcal{Q}' \leftarrow \mathcal{P}(\mathcal{Q})$
2. $\Sigma' \leftarrow \Sigma \setminus \{\epsilon\}$
3. $\delta' \leftarrow \{(q, a, r) \mid (q \in \mathcal{Q}') \wedge (a \in \Sigma') \wedge (r \leftarrow \bigcup_{s \in q} \delta(s, a))\}$
4. $q_0' \leftarrow \{q_0\}$
5. $\mathcal{F}' \leftarrow \{q \in \mathcal{Q}' \mid q \cap \mathcal{F} \neq \emptyset\}$

Minimierung von Automaten

Problem

Die Überführungsfunktion δ eines **DEA** ist nicht surjektiv.
(Es gibt Zustände, die nicht erreicht werden können.)

Eliminierung unnützer Zustände

Die Markierungen der Transitionen sind nicht relevant, der Graph des **DEA** wird als einfacher Digraph betrachtet.
Die Menge der erreichbaren Zustände lässt sich durch eine Fixpunktiteration bestimmen.

1. $\mathcal{Q}_0 \leftarrow \{q_0\}$
2. $\mathcal{Q}_{i+1} \leftarrow \mathcal{Q}_i \cup \mathcal{N}(\mathcal{Q}_i)$

Die Iteration wird fortgesetzt bis $\mathcal{Q}_{i+1} = \mathcal{Q}_i$.

$\mathcal{N}(\mathcal{Q}_i)$ bezeichne dabei die mit $\mathcal{Q}_i$ adjazenten Zustände, also die Nachbarn von $\mathcal{Q}_i$.

Algebra für Reguläre Ausdrücke

Assoziativität

1. $(\mathcal{R}_1 \circ \mathcal{R}_2) \circ \mathcal{R}_3 = \mathcal{R}_1 \circ (\mathcal{R}_2 \circ \mathcal{R}_3)$
2. $(\mathcal{R}_1 + \mathcal{R}_2) + \mathcal{R}_3 = \mathcal{R}_1 + (\mathcal{R}_2 + \mathcal{R}_3)$

Kommutativität

1. $\mathcal{R} \circ \epsilon = \epsilon \circ \mathcal{R} = \mathcal{R}$
2. $\mathcal{R} \circ \emptyset = \emptyset \circ \mathcal{R} = \emptyset$
3. $\mathcal{R} + \epsilon = \epsilon + \mathcal{R}$
4. $\mathcal{R} + \emptyset = \emptyset + \mathcal{R} = \mathcal{R}$

Distributivität

1. $(\mathcal{R}_1 + \mathcal{R}_2) \circ \mathcal{R}_3 = \mathcal{R}_1 \circ \mathcal{R}_3 + \mathcal{R}_2 \circ \mathcal{R}_3$

Algebra für Reguläre Ausdrücke

Idempotenz

1. $(\mathcal{R}^*)^* = \mathcal{R}^*$
2. $(\mathcal{R}^+)^* = \mathcal{R}^*$
3. $(\mathcal{R}^*)^+ = \mathcal{R}^*$
4. $\epsilon^* = \epsilon^+ = \epsilon$
5. $\mathcal{R}^* \circ \epsilon = \mathcal{R}^*$
6. $\mathcal{R}^* \circ \mathcal{R}^* = \mathcal{R}^*$
7. $\mathcal{R}^* + \mathcal{R}^* = \mathcal{R}^*$
8. $\mathcal{R}^+ \circ \mathcal{R}^+ = \mathcal{R} \circ \mathcal{R} \circ \mathcal{R}^*$
9. $\mathcal{R}^+ + \mathcal{R}^+ = \mathcal{R}^+$
10. $\mathcal{R}^* \circ \mathcal{R}^+ = \mathcal{R}^+ \circ \mathcal{R}^* = \mathcal{R}^+$

Chomsky Hierarchie

Sprachklassen

Die Menge aller Sprachen wurde durch Chomsky eingeteilt in die Chomsky-Hierarchie mit den 4 Sprachklassen

- Typ-3 : Reguläre Sprachen $\mathcal{L}_3$
- Typ-2 : Kontext-Freie Sprachen $\mathcal{L}_2$
- Typ-1 : Kontext-Sensitive Sprachen $\mathcal{L}_1$
- Typ-0 : Rekursiv Aufzählbare Sprachen $\mathcal{L}_0$

Chomsky Hierarchie

$$\mathcal{L}_3 \subsetneq \mathcal{L}_2 \subsetneq \mathcal{L}_1 \subsetneq \mathcal{L}_0$$

Frage

Wie lässt sich feststellen, ob

$$\mathcal{L} \in \mathcal{L}_3?$$

Antwort

Eine Sprache ist vom Typ-3, wenn sie von einem endlichen Automaten akzeptiert wird.
Da Endliche Automaten, Reguläre Ausdrücke und Typ-3-Grammatiken äquivalent sind (sie lassen sich ineinander überführen) ist für den Nachweis $\mathcal{L} \in \mathcal{L}_3$ ein Endlicher Automat $\mathcal{A}$, ein Regulärer Ausdruck $\mathcal{R}$ oder eine Typ-3-Grammatik $\mathcal{G}$ mit

$$\mathcal{L}_{\mathcal{A}} = \mathcal{L} \text{ bzw. } \mathcal{L}_{\mathcal{R}} = \mathcal{L} \text{ bzw. } \mathcal{L}_{\mathcal{G}} = \mathcal{L}$$

anzugeben.

Frage

Wie lässt sich feststellen, ob

$$\mathcal{L} \notin \mathcal{L}_3?$$

Antwort

Mit Hilfe des Pumping-Lemmas.
Wenn die Bedingungen des Pumping-Lemmas nicht erfüllt sind, so kann die betroffene Sprache nicht regulär sein.

$$\overline{PL(\mathcal{L})} \rightarrow \mathcal{L} \notin \mathcal{L}_3.$$

Das Pumping-Lemma

Sei $\mathcal{L} \subseteq \Sigma^*$ eine reguläre Sprache.
Dann gibt es eine natürliche Zahl $p \in \mathbb{N}$ derart, dass sich jedes Wort $x \in \mathcal{L}$ mit $|x| \geq p$ zerlegen lässt in drei Teilworte

$$x = uvw$$

mit

1. $|v| > 0$
2. $|uv| \leq p$
3. $\bigwedge_{i \in \mathbb{N}} uv^i w \in \mathcal{L}$

Vorsicht!

Es gilt $\mathcal{L} \in \mathcal{L}_3 \rightarrow PL(\mathcal{L})$
Es gilt **nicht** $PL(\mathcal{L}) \rightarrow \mathcal{L} \in \mathcal{L}_3$

Exkurs

Problem

Es gibt Sprachen mit $\mathcal{L} \notin \mathcal{L}_3$, für die aber das Pumping-Lemma erfüllt ist.

Nerode-Relation

Seien $x, y \in \Sigma^*, \mathcal{L} \subseteq \Sigma^*, \mathcal{R} \subseteq \Sigma^* \times \Sigma^*$.

$$\mathcal{R}_{\mathcal{N}} : x \sim y \Leftrightarrow \bigwedge_{z \in \Sigma^*} (xz \in \mathcal{L}) \Leftrightarrow (yz \in \mathcal{L})$$

Satz von Myhill und Nerode

Wenn der Index der Nerode-Relation endlich ist, also wenn es nur endlich viele Äquivalenzklassen bezüglich $\mathcal{R}_{\mathcal{N}}$ gibt, so gilt $\mathcal{L} \in \mathcal{L}_3$.

Minimierung Endlicher Automaten

Problem

Die Darstellung eines Endlichen Automaten ist nicht eindeutig, es gibt (abzählbar) unendlich viele Automaten, die alle dieselbe Sprache akzeptieren.
Außerdem kann ein Automat Zustände enthalten, die nutzlos sind oder aber welche, die äquivalent zueinander sind.

Automaten-Minimierung

Bei der Minimierung sollen unnütze Zustände entfernt und äquivalente Zustände zu Äquivalenzklassen zusammengefasst werden.
Ausgehend von einem Automaten $\mathcal{A}$ entsteht der sogenannte Äquivalenzklassenautomat $\mathcal{A}_\sim$ mit $\mathcal{A} \simeq \mathcal{A}_\sim$.

Ablauf der Minimierung

Sei $\mathcal{A}$ ein DEA.

1. Entfernen unerreichbarer Zustände
2. Bilden der Äquivalenzklassen

Definition der Äquivalenzklassen

$$\mathcal{R} \subseteq \mathcal{Q} \times \mathcal{Q}$$

mit

$$\mathcal{R} : q_i \sim q_j \Leftrightarrow \bigwedge_{w \in \Sigma^*} [\delta^*(q_i, w) \in \mathcal{F}] \Leftrightarrow [\delta^*(q_j, w) \in \mathcal{F}].$$

Anmerkung

Die entstehenden Äquivalenzklassen stimmen mit denen der Nerode-Relation überein.

Wen es interessiert ...

Nerode-Relation

Seien $x, y \in \Sigma^*, \mathcal{L} \subseteq \Sigma^*, \mathcal{R} \subseteq \Sigma^* \times \Sigma^*$.

$$\mathcal{R}_{\mathcal{N}} : x \sim y \Leftrightarrow \bigwedge_{z \in \Sigma^*} (xz \in \mathcal{L}) \Leftrightarrow (yz \in \mathcal{L})$$

Definition der Äquivalenzklassen

$$\mathcal{R} \subseteq \mathcal{Q} \times \mathcal{Q}$$

mit

$$\mathcal{R} : q_i \sim q_j \Leftrightarrow \bigwedge_{w \in \Sigma^*} [\delta^*(q_i, w) \in \mathcal{F}] \Leftrightarrow [\delta^*(q_j, w) \in \mathcal{F}].$$

Definition der Äquivalenzklassen

$$\mathcal{R} : q_i \sim q_j \Leftrightarrow \bigwedge_{w \in \Sigma^*} [\delta^*(q_i, w) \in \mathcal{F}] \Leftrightarrow [\delta^*(q_j, w) \in \mathcal{F}].$$

Problem

Es muss für alle Wörter aus Σ^* geprüft werden, ob die Bedingung erfüllt ist.
Das ist aber nicht möglich.
Man verschafft sich Abhilfe dadurch, dass man die negierte Aussage verwendet

$$\mathcal{R} : q_i \nsim q_j \Leftrightarrow \bigvee_{w \in \Sigma^*} [\delta^*(q_i, w) \in \mathcal{F}] \wedge [\delta^*(q_j, w) \notin \mathcal{F}].$$

Ein solches w nennen wir einen Zeugen $\rightarrow$ Tafel.