Vorlesungsskript

zur Vorlesung „Theoretische Informatik“

Vorlesung vom 09.06.2000

Von:
Jens Habermann / 703333
Jan Höhle / 703489

Wir beschäftigen uns mit folgender Fragestellung:
Kann man maschinell entscheiden ob für einen gegebenen regulären Ausdruck R gilt:

o(R)=X*.

Das Problem beschreibt das Maximalitätsproblem bzw. Mehrheitsproblem. Das Problem ist algorithmisch lösbar (siehe Übung).

<u>Systematische Konstruktion von Akzeptoren</u>

In der letzten Vorlesung haben wir gezeigt, wie man zu einen beliebigen Automaten, einen regulären Ausdruck konstruiert, der die gleiche Sprache beschreibt. Im folgenden werden wir zu einem beliebigen regulären Ausdruck, einen deterministischen endlichen Automaten konstruieren, der genau die Sprache des regulären Ausdrucks akzeptiert.

<u>1. Induktive Vorgehensweise:</u>

Idee: Wir geben für jeden regulären Teilausdruck einen endlichen Automaten an. Damit können wir zu jedem regulären Ausdruck induktiv einen endlichen Automaten erzeugen. Die Automaten besitzen jeweils nur einen Anfangszustand und einen Endzustand und es gibt keine Kanten die zum Anfangszustand hinführen oder vom Endzustand wegführen.

Beispiel: $(a \cup b)^* \cdot c$

Der reguläre Ausdruck wird von der innersten Klammer nach außen hin zum Automaten entwickelt. Also modelliere ich erst a, dann b und dann $a \cup b$, usw.

Für den regulären Ausdruck $\emptyset$ verwenden wir den Automaten aus Abbildung 1:

Abb. 1: Automat für die Sprache $\emptyset$

Für den regulären Ausdruck ε verwenden wir den Automaten aus Abbildung 2:

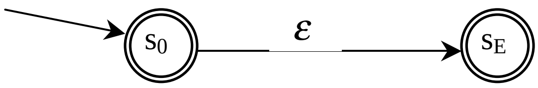


Abb. 2: Automat für die Sprache $\{\varepsilon\}$

Für den regulären Ausdruck x∈ X (ein Alphabet) verwenden wir den Automaten aus Abbildung 3:

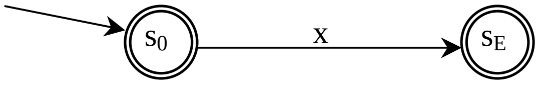


Abb. 3: Automat für die Sprache {x}

Für den regulären Ausdruck $R \cdot R'$ verwenden wir den Automaten aus Abbildung 4:

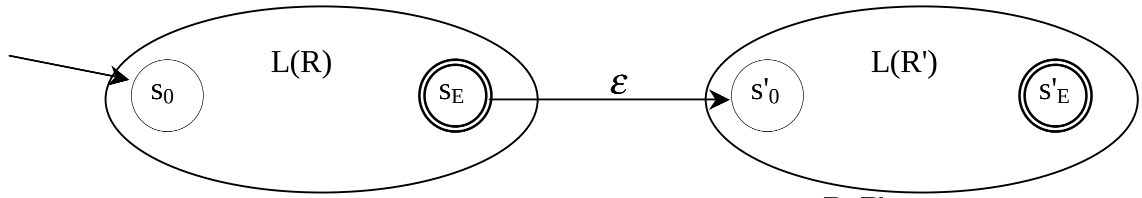


Abb. 4: Automat für die Sprache $R \cdot R'$

Für den regulären Ausdruck $R \cup R'$ verwenden wir den Automaten aus Abbildung 5:

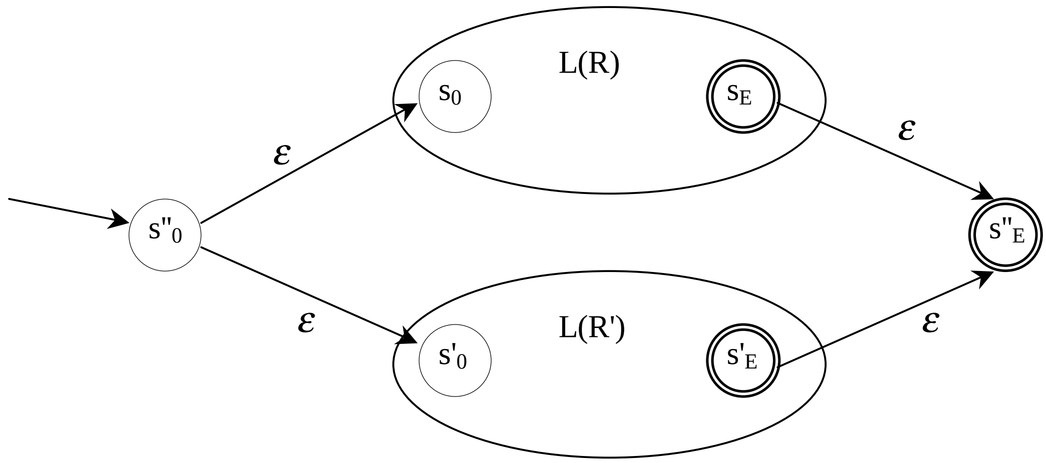


Abb. 5: Automat für die Sprache $R \cup R'$

Für den regulären Ausdruck R* verwenden wir den Automaten aus Abb.6.

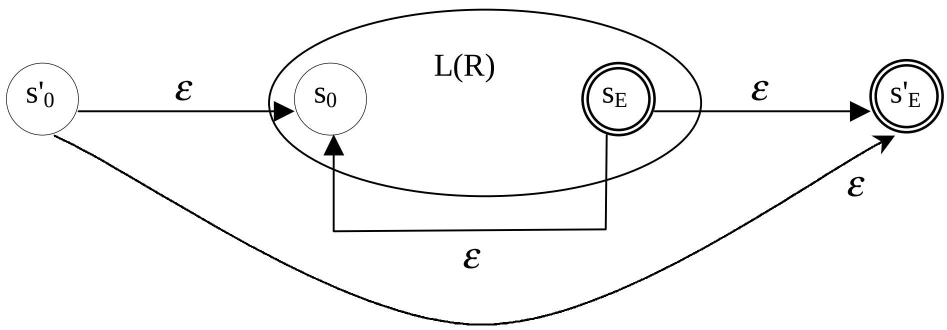


*Abb.6: Automat für die Sprache R**

Für den regulären Ausdruck X*\R verwenden wir den Automaten aus Abbildung 7:

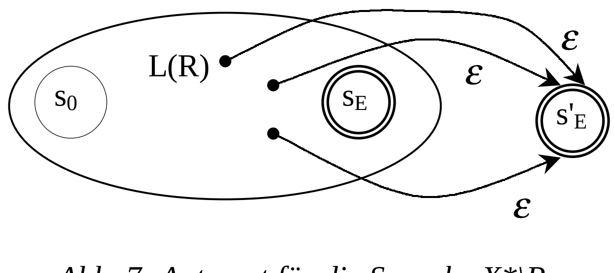
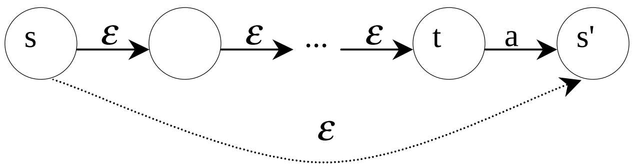


Abb. 7: Automat für die Sprache X\R*

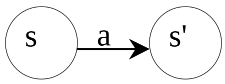
2. Entfernen der ε -Übergänge

Nun werden die ε-Übergänge gelöscht um einen vollständigen Automaten zu erhalten. Dies ist nötig, um den 3. Schritt möglich zu machen.
Dazu müssen folgende Schritte abgearbeitet werden:

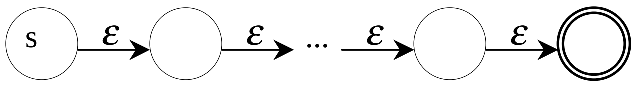
a) Fall es den Übergang



gibt, dann ergänze



b) Falls es den Übergang



gibt, dann mache s zum Endzustand.

c) Lösche alle ε -Übergänge.

3. Deterministischen Automaten konstruieren

Wie wir bereits wissen, können wir aus einem nicht deterministischen vollständigen Automaten einen deterministischen endlichen Automaten gewinnen. s. o.

4. Zustandsminimierung

Die Zustandsminimierung wird erst in den nächsten Vorlesungen erläutert.
Beispielprogramme für die Generierung von zustandsminimalen Automaten aus reg. Ausdrücken sind lex und yacc.

4.4 Algebraische Charakterisierung regulärer Sprachen

Wir erkennen folgende Charakterisierungen:

$$\otimes\left\{\begin{array}{l}\text{- endliche deterministische Automaten}\\ \text{- endliche nicht - deterministische Automaten}\end{array}\right\}\text{Automaten – bezogen}\ \Big]\text{"erkennen Sprachen"}$$

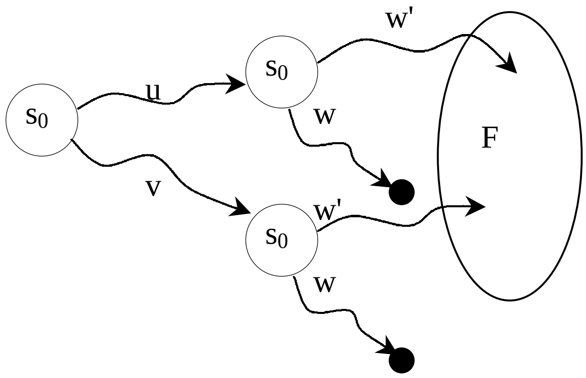
$$\text{- reguläre Ausdrücke}\}\text{Sprachbezogen}\ ]\text{"erzeugen Sprachen"}$$

$$\text{- algebraische Charakteristik}\ ]\text{"erzeugen Sprachen"}$$

$$\otimes\ (\text{später : grammatikalische Charakterisierung})$$

Def. R: Sei $L \subseteq X^*$ eine Sprache. Die Relation $\equiv_L$ definiert durch
$u \equiv_L v \Leftrightarrow \forall w \in X^* (uw \in L \Leftrightarrow vw \in L)$ heißt Nerode-Äquivalenz zur Sprache L.

[u und v können bei beliebiger Konkatenation mit $w \in X^*$ zugleich zu Worten $\in L$ bzw. $\notin L$ fortgesetzt werden.]



Bemerkung 1: $\equiv_L$ ist eine Äquivalenzrelation, denn:
$\equiv_L$ ist reflexiv $\forall u \in X^* (u \equiv_L u)$
$\equiv_L$ ist symmetrisch $\forall u, v \in X^* (u \equiv_L v \Leftrightarrow v \equiv_L u)$
$\equiv_L$ ist transitiv $\forall u, v, w \in X^* (u \equiv_L v \wedge v \equiv_L w \Rightarrow u \equiv_L w)$

Bemerkung 2: $\equiv_L$ ist verträglich mit Rechtskonkatenation:
$\forall u, v, w \in X^* (u \equiv_L v \Rightarrow uw \equiv_L vw)$

Def. S: Sei für $u \in X^*$:
$[u] := \{v \in X^* | u \equiv_L v)$ die Äquivalenzklasse von u bezüglich $\equiv_L$
$K_L := \{[u] | u \in X^*\}$ ist definiert als die Menge der Äquivalenzklassen. $|K_L|$ ist der Index von $\equiv_L$

Erklärung: Die Äquivalenzklasse von u gibt an, welche Wörter aus X* mit u in Relation stehen.

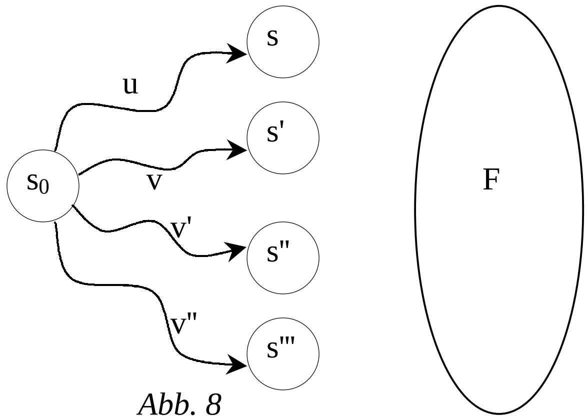


Abb. 8

Die Abbildung 8 zeigt, wie man einen Automaten zur Äquivalenzklasse von u generiert. Führt u zu einem Endzustand, so gilt dies auch für alle anderen Wörter, die in der Klasse liegen. Gilt v, v', v''$\in$ [u], so sagt man auch, das funktionale Verhalten von s, s', s'' und s''' ist gleich.

Satz T: L ist genau dann regulär, wenn der Index von $\equiv_L$ endlich ist. (Satz von Myhill/Nerode)

Beweis: „$\Rightarrow$“: Sei L regulär, zeige $|K_L|$ endlich. $L \subseteq X^*$. Sei A=(X, S, δ , s_0, F) ein det. endlicher Akzeptor mit L=L(A). Betrachte 2 Wörter $u, v \in X^*$ mit
δ (s_0,u)=s=δ (s_0,v).
Für alle $w \in X^*$ gilt jetzt:
δ (s,w)=δ (s_0,uw) =δ (s_0,vw), d.h. $uw \in L \Leftrightarrow vw \in L$.
Wenn also δ (s_0,u) =δ (s_0,v), dann folgt u $\equiv_L$ v und [u]=[v].
Folglich ist die Zahl der Äquivalenzklassen höchstens so groß, wie die Zahl der Zustände |S|, also ist der Index von $\equiv_L$ $\leq$|S|, also endlich.

„$\Leftarrow$“: Sei L eine Sprache mit endl. Index. Dann ist nach Definition S
$K_L = \{[u_1],...,[u_k]\}$ für geeignete Wörter $u_1,...,u_k \in X^*$. Definiere nun einen endlichen Akzeptor, der L akzeptiert.

Idee: Verwende Äquivalenzklassen als Zustände. Sei also A=(X, S, δ , s_0, F) mit
$S = K_L, s_0 = [\varepsilon], F = \{[w] \mid w \in L\}, \delta : S \times X \to S$ mit $\delta([w], x) = [wx], \forall x \in X$.
Ist Def. sinnvoll? (Wohldefiniertheit)
Nachprüfen, dass A bzw. δ wohldefiniert sind. Was hier soviel heißt, wie: Egal, welchen Vertreter der Äquivalenzklasse man verwendet, das Ergebnis von δ muß eindeutig bestimmt sein.
Zeige also: $[v] = [u] \Rightarrow \delta([v], x) = \delta([u], x), \forall x \in X$.
Zum Beweis: $[v] = [u] \Rightarrow v \equiv_L u \overset{Def.R+Bem.2}{\Rightarrow} \forall x \in X (vx \equiv_L ux \Rightarrow [vx] = [ux])$.
Also ist δ wohldefiniert.

Zeige noch: L(A)=L
Sei $w \in L(A) \Rightarrow \delta(s_0, w) \in F \Rightarrow \delta([\varepsilon], w) \in F \Leftrightarrow [w] \in F \Leftrightarrow w \in L.$

Erklärung: Wir wissen, wir können zu jeder regulären Sprache einen endlichen deterministischen Automaten angeben.
„$\Rightarrow$“: Die Betrachtung der Äquivalenzklassen eines endlichen Automaten gibt Auskunft darüber, wie viele Äquivalenzklassen die reguläre Sprache besitzt. Die Anzahl der Automatenzustände ist größer oder gleich der Anzahl der Äquivalenzklassen und damit in jedem Fall endlich.
„$\Leftarrow$“: Die Anzahl der Zustände ist minimal, wenn wir zur Konstruktion die Äquivalenzklasse der Nerode-Äquivalenz nutzen. Damit ist die Anzahl der Automatenzustände gleich der Anzahl der Äquivalenzklassen. Folglich ist der Automat endlich, bzw. die Sprache regulär, wenn der Index der Nerode-Äquivalenz endlich ist.