

Automatentheorie und Formale Sprachen

Grundlagen:

Kartesisches Produkt: $X \times Y = \{(x, y) | x \in X \wedge y \in Y\}$

Es gilt: $|X \times Y| = |X| * |Y|$ für X, Y endlich

Halbgruppe: (H, \circ) mit $x \circ e = e \circ x = x$

Monoid: (M, \circ, e) , abgeschlossen, assoziativ, neutrales e

Produktmonoid: M, N Monoide: $(M \times N, [\circ, *], (e, 1))$ ist ein Monoid

Produktmorphismus: Wenn $h_M: (X, \Delta, I) \rightarrow (M, \circ, e)$, $h_N: (Y, \Delta, I) \rightarrow (N, *, 1)$ Monoidmorphismen,
dann auch $h_M \times h_N: X \times Y \rightarrow M \times N$, $x \mapsto (h_M(x), h_N(y))$

Potenzmenge: 2^X Menge der Teilmengen

Alle behandelten Funktionen sind total, jedem X wird genau ein Y zugeordnet

Äquivalenzrelation: $X \times X$ mit Reflexiv, transitiv, symmetrisch

Homomorphismus: $h(x \circ y) = h(x) * h(y)$ mit * Verknüpfung von y-Gruppe

Isomorphismus: Homomorphismus mit $h(e) =$ neutrales Element in $(\{h(x) | x \in H\}, *)$

Diagonale: $\Delta_X = \{(x, x) | x \in X\}$

transitive Hülle: $R^+ = \bigcup_{n \geq 1} R^n$ (Bei R in REG: Halbgruppe)

reflexiv transitive Hülle: $R^* = \bigcup_{n \geq 0} R^n$ (Bei R in REG: Monoid)

Spiegeloperation: w^R rückwärts

Typ einer algebraischen Struktur: (F, σ) mit F: Funktionensymbole, $\sigma: F \rightarrow \mathbb{N}$ liefert Stelligkeit

Algebraische Struktur vom Typ (F, σ) : $A = (A, F)$, $A \neq 0$, $F = \{f_A | f \in F\}$

Terme über $A = (A, F)$:

Jedes $a \in A$ ist ein Term

Sind $t_1 \dots t_n$ Terme und $f \in F$ mit $\sigma(f) = n$, so ist $f(t_1 \dots t_n)$ ein Term

$A^n \rightarrow A$ heißt n-stellige Operation

0-stellige Operationen sind Konstanten

Sprachen:

Alphabet: endliche, nicht-leere Menge aus Buchstaben/Zeichen Σ

Wort: $w \in \Sigma^n$ der Länge n

Konkatenation: Aneinanderhängen von Wörtern

Sprache: $L \subseteq \Sigma^*$

Reguläre Sprachen: L regulär, gdw. $\exists (M, \circ, e)$ Monoid, $h: (\Sigma^*, *, \lambda) \rightarrow (M, \circ, e)$ Monoidmorphismus,

endliche Menge $F \subseteq M$ mit $L = \{w \in \Sigma^* | h(w) \in F\}$

L ist regulär, gdw. L von DEA akzeptiert wird.

Eine Menge von Sprachen heißt Sprachfamilie

REG: Reguläre Sprachen

Abschlusseigenschaften regulärer Sprachen:

Vereinigung, Schnitt, Komplement, Konkatenation, Kleene-Stern abgeschlossen

Sei $(\Sigma^*, *, \lambda) \rightarrow (M, \circ, e)$ Monoidmorphismus, dann definiere $x \equiv_h y$ gdw. $h(x) = h(y)$

$x \equiv_h y$ ist eine Äquivalenzrelation auf Σ^*

$L \subseteq \Sigma^*$ trennt zwei Wörter $x, y \in \Sigma^*$ gdw $|\{x, y\} \cap L| = 1$

Zwei Wörter u, v heißen kongruent modulo L $u \equiv_L v$ wenn für jedes $w \in \Sigma^*$ die Sprache L die Wörter uw und vw nicht trennt: $(\forall w \in \Sigma^*: uw \in L \leftrightarrow vw \in L)$

heißt auch Myhill-Nerode-Äquivalenz (Ist Äquivalenzrelation über L)

Ist Rechtskongruent: $u \equiv_L v \rightarrow \forall x \in \Sigma^*: ux \equiv_L vx$

Es gilt $u \equiv_h v$ so auch $u \equiv_L v$ falls h der Monoidmorph. vom L beschriebenen Monoid

Ist L regulär, so hat \equiv_L endlich viele Äquivalenzklassen

Teilwort: $u: x \in \Sigma^*\{u\}\Sigma^*$ Präfix: $u: x \in \{u\}\Sigma^*$ Suffix: $u: x \in \Sigma^*\{u\}$

Ein Teilwort/Präfix/Suffix heißt echt, wenn $l(u) < l(x)$

Deterministischer endlicher Automat:

$A = (Q, \Sigma, \delta, q_0, F)$ Q: endliche Zustandsmenge, Σ : endliches Eingabealphabet

$\delta: Q \times \Sigma \rightarrow Q$ Überführungsfunktion

q_0 Anfangszustand

$F \subseteq Q$ Endzustände

Konfiguration: $C = Q \times \Sigma^*$

Übergangsfunktion: $(q, w) \vdash_A (q', w')$ gdw $\exists a \in \Sigma: w = aw', q' = \delta(q, a)$

Akzeptierte Sprache: $L(A) = \{w \in \Sigma^* \mid \exists q \in F: (q_0, w) \vdash_A^* (q, \lambda)\}$

Schlingenlemma: A DEA, sei $X = \{a \in \Sigma: (q, a) \vdash_A (q, \lambda)\}$, dann gilt: $X^* \subseteq \{w \in \Sigma^* \mid (q, w) \vdash_A^* (q, \lambda)\}$

Um zu beweisen, dass ein Automat eine Sprache akzeptiert muss so bewiesen werden:

1. $L(A) \subseteq L$, 2. $L(A) \supseteq L$

DEA-Sprachen sind komplementabgeschlossen (Endzustandsmengenkomplement)

Automatenmorphismus: Homomorph. $f: Q_1 \rightarrow Q_2$ gdw.

$\forall a \in \Sigma \forall q \in Q_1: f(\delta_1(q, a)) = \delta_2(f(q), a)$

$f(q_{01}) = q_{02}$

$\forall q \in Q_1: F_1 \Leftrightarrow f(q) \in F_2$

Wenn es einen Automatenmorph. zwischen den DEA A_1 und A_2 gibt, ist $L(A_1) = L(A_2)$

Es gibt zu jeder REG genau einen Minimalautomaten:

$Q = \{[x_1], \dots, [x_k]\}, q_0 = [y], F = \{[x_i] \mid x_i \in L\}, \delta([x], a) = [xa]$

Konstruktion Minimalautomat:

1. Bestimme alle von q_0 erreichbaren Zustände, seien E_i Zustände in $\leq i$ erreichbar

a) $E_0 = \{q_0\}, E_{-1} = \emptyset$

b) Wiederhole: $E_{i+1} = E_i \cup \{\delta(q, a) \mid q \in E_i \setminus E_{i-1}, a \in \Sigma\}$

c) $E = E_i$ wenn erstmals $E_i = E_{i+1}$

d) Entferne Zustände $Q \setminus E$ aus A

2. Bestimme \equiv_A mit Markierungsalgorithmus:

a) Tabelle aller ungeordneten Zustandspaare $\{q, q'\}$ mit $q \neq q'$

b) Markiere alle Paare als nicht-äquivalent, wenn $|\{q, q'\} \cap F| = 1$

c) Wiederhole: Für jedes nicht markierte Paar und jedes $a \in \Sigma$

Teste, ob $\{\delta(q, a), \delta(q', a)\}$ markiert, wenn ja, markiere $\{q, q'\}$

d) Alle nichtmarkierten Paare sind äquivalent

Ein DEA ist minimal wenn:

Es keine nichterreichbaren Zustände gibt

Er keine äquivalenten Zustände hat: $(q, w) \vdash_A^* (p, \lambda) \Leftrightarrow (q', w) \vdash_A^* (p', \lambda)$

Eigenschaften Äquivalenter Zustände q, q' :

$\delta(q, a) \cong \delta(q', a)$ denn $(\delta(q, a), w) = (q, aw), (q', aw) = (\delta(q', a), w)$

$q \in F \Leftrightarrow q' \in F$

Die Nicht-Trennbarkeit ist eine Äquivalenzrelation auf Q

Leerheitsproblem: $L(A) = \emptyset$ gdw. E (Erreichbare) keine Endzustände enthält

Teilmengen/Äquivalenzproblem: Bestimme für $L(A)$ und $L(A') L(A'')=L(A)\backslash L(A')$ (Produktautomaten)
Entscheide Leerheitsproblem für A''

Endlichkeitsproblem: Wenn der Automat eine Schleife hat ist die Sprache unendlich
Wenn q von q aus erreichbar ist

Nichtdeterministischer endlicher Automat:

$A = (Q, \Sigma, \delta, Q_0, F)$, Q_0 Anfangszustände

$(q, w) \vdash_A (q', w')$, wenn es ein Zeichen a gibt mit $w = aw'$ und $(q, a, q') \in \delta$

$L(A) = \{w \in \Sigma^* | \exists q_0 \in Q_0, q \in F: (q_0, w) \vdash_A^* (q, \lambda)\}$

Jede endliche Sprache ist eine NEA-Sprache

NEAs erlauben λ -Übergänge, kann in NEA ohne λ -Übergänge umgewandelt werden.

Skelettautomat: $w = a_{i,1} \dots a_{i,l(w)}$ $1 \leq i \leq M$

$$Q = \{(i, j) | 1 \leq i \leq M, 0 \leq j \leq l(w)\}$$

$$Q_0 = \{(i, 0) | 1 \leq i \leq M\}, F = \{(i, l(w)) | 1 \leq i \leq M\}$$

$$\delta = \{(i, j), a(i, j+1) | 1 \leq i \leq M, 0 \leq j \leq l(w), a_{i,j+1} = a\}$$

Reguläre Ausdrücke:

Induktive Definition:

1. \emptyset und a sind RA für jedes $a \in \Sigma$
2. Ist R ein RA, so auch $(R)^*$
3. Sind R_1 und R_2 RA, so auch R_1R_2 und $(R_1 \cup R_2)$

Sprache:

1. $L(\emptyset), L(a)=\{a\}$ sind Sprachen
2. Ist R ein RA, setze $L((R)^*)=(L(R))^*$
3. Sind R_1, R_2 RA, setze $L(R_1R_2) = L(R_1)L(R_2)$ und $L((R_1 \cup R_2)) = L(R_1) \cup L(R_2)$

RA-Sprachen sind regulär. Beweis:

1. Endliche Sprachen sind regulär
2. REG ist Kleene-Stern abgeschlossen
3. REG ist Vereinigung/Konkatenation abgeschlossen

Jede reguläre Sprache lässt sich als RA darstellen

Pumping-Lemma:

Zu jeder REG gibt es eine Zahl $n > 0$, sodass jedes Wort $w \in L$ mit $l(w) \geq n$ als Konkatenation $w=xyz$ dargestellt werden kann, mit:

1. $l(y) > 0$
2. $l(xy) \leq n$
3. $\forall i \geq 0: xy^i z \in L$

Kontextfreie Sprachen:

$G = (\Sigma, N, R, S)$ mit Terminalalphabet, Nonterminalalphabet, Regeln ($R \subset N \times (\Sigma \cup N)^*$),
Statsymbol

Ableitungsrelation: $\Rightarrow_G: \exists (A \rightarrow y) \in R: u = xAz, v = xyz$ mit $u, v \in (\Sigma \cup N)^*$

Die erzeugte/abgeleitete Sprache einer Grammatik: $L(G) := \{w \in \Sigma^* | S \xrightarrow{*} w\}$
(Kontextfreie Sprache KF)

$REG \subset KF$

Rechtslinear: Wenn alle rechten Regelseiten die Form Az oder z haben, z Terminalwort, A Nichtterminalzeichen

L regulär $\Leftrightarrow L$ durch rechtslineare kfG erzeugbar

Regeln können links oder rechts abgeleitet werden (linkes oder rechtes Nichtterminal ersetzt)

Baumautomat:

$$A = (Q, (\mathbf{F}, \sigma), Q_f, \Delta)$$

Q : Zustandsalphabet

Q_f : Endzustandsmenge

Δ : $(f, q_1 \dots q_n, q)$ mit $f \in \mathbf{F}$, $\sigma(f) = n$, $q_1 \dots q_n, q \in Q$