Theoretische Informatik
Vorlesungsmitschrift

Sebastian T. Hafner
sonix@uni-saarland.info
Christian Holler
decoder@uni-saarland.info

1. Dezember 2004
last compile: 7. Dezember 2004

Inhaltsverzeichnis

1 Rechnen an sich

- Was ist „Rechnen“?
- Welche Rechenmechanismen gibt es?
- Kann man „alles“ „berechnen“?
- Wie „teuer“ ist Rechnen?

2 Rechenmodelle / Rechensituationen

2.1 PC / Mac

ZEICHNUNG CHRISTIAN HOLLER 1

Auf bestimmte Eingaben sollen bestimmte Ausgaben erfolgen.

2.2 AIRBUS

viele vernetzte Computereinheiten

Eingaben:	Steuerungen durch Besatzung
	Sensoren
	Temperatur
	GPS
	Luftgeschwindigkeit
	Positionsanzeige / Kompass
Ausgabe:	Informationen für Besatzung
	Aktuatoren

2.3 „Einfachste Situation“: Theoretitischer PC

ZEICHNUNG CHRISTIAN HOLLER 2

soll	Funktion F: Zeichenketten	⟵ Zeichenketten berechnen
	Eingabe: s	Ausgabe F(s) $\{0,1\}$ (Ja / Nein)

2.4 EINSCHUB: Zeichenketten

Alphabet:	Σ	endliche Menge
String / Zeichenkette		endliche Folge von Elementen von Σ
		z.B. $v = v_1 v_2 \dots v_k \; v_i \in \Sigma \forall i$
		$\underbrace{\lvert v \rvert}_{\text{Länge}} = k$

Σ^k Menge der Strings der Länge k

$$\{\}^k = \{\} \quad \forall k \ ?$$

Σ^0

$$\Sigma^0 = \left\{ \underbrace{\epsilon}_{leererString} \right\}$$

Σ^* Menge der Strings über Σ (mit endlicher Länge)

$$\Sigma^* = \cup_{k \geq 0} \Sigma^k$$

2.4.1 Konkatenation

$$\begin{aligned} u, v &\in \Sigma^* \\ u &\in \Sigma^l \\ v &\in \Sigma^k \\ u\epsilon &= u \\ \epsilon u &= u \\ uv &= u_1 u_2 \dots u_l v_1 \dots v_k \end{aligned}$$

$$\begin{aligned} v &\in \Sigma^* i \in \mathbb{N} = \{0, 1, \dots\} \\ v^0 &= \epsilon \\ v^i &= v v^{i-1} \\ L \subset \Sigma * \dots &\text{„Sprache über } \Sigma\text{“} \end{aligned}$$

2.4.2 „Maschine definiert Sprache“

$$\{s \in \Sigma^* \mid F(s) = JA\}$$

3 Welche Sprachen können von welchen Arten von „Automaten“ berechnet werden?

3.1 3 Arten von Automaten

1. Endliche Automaten
2. Keller Automat

3. Turing Maschine

3.2 Endlicher Automat

Eingabeband

a	b	b	a	a	a	

ZEICHNUNG CHRISTIAN HOLLER 3

endlich viele Steuerungsregeln folgender Form:

Wenn Zustand $= q$
$\wedge$ Eingabekopf liest Zeichen a
dann:
1) Rücke Eingabekopf um eine Zelle nach rechts
2) wechsle in Zustand q'

$s \in Q$ Startzustand
$F \subset Q$ Endzustände

3.2.1 Anfangskonfiguration

Maschine ist im Startzustand s
Eingabekopf ganz links

3.2.2 Endkonfiguration

Maschine ist in einem der Endzustände
Eingabekopf ganz rechts

Maschine akzeptiert einen Engabestring u, wenn aus der Startkonfiguration durch wiederholtes Anwenden von Regeln eine Endkonfiguration erreicht werden kann.

3.3 Beispiel

3.3.1 Konfiguration

$$\Sigma = \{a, b\} \qquad Q = \{\underbrace{0}_{=s}, 1, 2\} \qquad F = \{2\}$$

3.3.2 Regeln

derzeitiger Zustand	gelesenes Zeichen	neuer Zustand
0	a	1
0	b	0
1	a	2
1	b	0
2	a	2
2	b	0

3.3.3 ungültiger String

a	b	b	a		
↑	↑	↑	↑	↑	
0	1	0	0	$\underline{1}$	ungültiger Endzustand

3.3.4 gültiger String

a	b	a	a	
↑	↑	↑	↑	↑
0	1	0	0	2

Das Beispiel akzeptiert alle Strings über $\{a, b\}$, die in aa enden.

3.4 Nicht deterministische Maschine

d.i. mehr als eine Regel in gleichen Zuständen anwendbar.

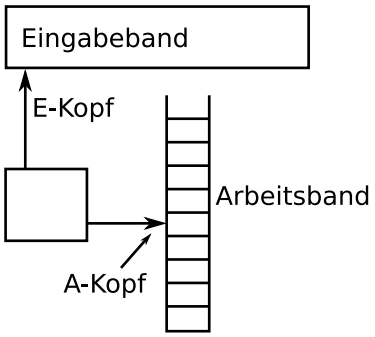
	derzeitiger Zustand	gelesenes Zeichen	neuer Zustand
α	0	a	0
β	0	b	0
γ	0	a	1
δ	1	a	2

a	b	a	a	
↑	↑	↑	↑	↑
0	1	0	0	2
α	β	γ	δ	

3.5 Kellerautomat

Eingabeband

a	b	b	a	a	a	



Q	Zustandsmenge
$s \in Q$	Startzustand
$F \subset Q$	Endzustand

3.5.1 Regeln:

Wenn Zustand = q
$\wedge$ Eingabekopf liest a
$\wedge$ Arbeitskopf liest A

dann:
1) gehe in Zustand q'
2) Bewege Eingabekopf
i) rechts
ii) nicht
3) Arbeitskopf: i) ersetze A durch A'
ii) Kopf nach unten, A gelöscht
iii) Kopf nach oben, schreibe A'

3.5.2 Anfangskonfiguration

Zustand = s
Eingabekopf ganz links
Arbeitsband

ZEICHNUNG CHRISTIAN HOLLER 5

3.5.3 Endkonfiguration

Maschine in einem der Endzustände
Eingabekopf ganz rechts

Maschine akzeptiert einen Engabestring u, wenn aus der Startkonfiguration durch wiederholtes Anwenden von Regeln eine Endkonfiguration erreicht werden kann.

$$\{a^n b^n \mid n \geq 1\}$$

Deterministisch immer höchstens 1 Regel anwendbar

Nicht Deterministisch es können mehrere Regeln anwendbar sein

3.6 Keller Maschine

$BILD:$

Zustand
Eingabekopf $\longrightarrow$ EK $\longrightarrow$ $\bullet$
Arbeitskopf

3.6.1 Beispiel

$u\ in\{a,b\}$
$u = u_1 \dots u_n$
$u^R = u_n \dots u_1$
$u\$u^R\$$

derzeitiger Zustand	EK	AK	neuer Zustand	EK	AK
$\underbrace{0}_{\text{Anfangszustand}}$	a	Z	1	$\rightarrow$	$\uparrow$ A
0	b	Z	1	$\rightarrow$	$\uparrow$ B
1	a	A	1	$\rightarrow$	$\uparrow$ A
1	a	B	1	$\rightarrow$	$\uparrow$ A
1	b	A	1	$\rightarrow$	$\uparrow$ B
1	b	B	1	$\rightarrow$	$\uparrow$ B
1	\$	A	2	$\rightarrow$	A
1	\$	B	2	$\rightarrow$	B
2	a	A	2	$\rightarrow$	$\downarrow$
2	b	B	2	$\rightarrow$	$\downarrow$
2	\$	Z	$\underbrace{3}_{\text{Endzustand}}$	$\leftarrow$	$\downarrow$

3.7 Turing Maschine

$BILD$

3.7.1 Regeln

Wenn	Zustand = q		neuer Zustand q'	
	Eingabekopf liest a	dann	Eingabekopf	$\rightarrow, \bullet$
	Arbeitskopf liest A		Arbeitskopf	A',$\leftarrow, \bullet, \rightarrow$

DEA	deterministischer endlicher Automat
NEA	nichtdeterministischer endlicher Automat
DKA	deterministischer Keller Automat
NKA	nichtdeterministischer Keller Automat
DTM	deterministische Turing Maschine
NTM	nichtdeterministische Turing Maschine

3.8 Definition

Eine Sprache L ist eine DEA-Sprache, wenn es einen DEA gibt, der L akzeptiert.
und analog andere Sprachen.

3.9 Lemma

- Jede DEA-Sprache ist eine NEA-Sprache.
- Jede DKA-Sprache ist eine NKA-Sprache.
- Jede DTM-Sprache ist eine NTM-Sprache.

3.10 Lemma

- Jede DEA Sprache ist eine DKA-Sprache.
- Jede NEA Sprache ist eine NKA-Sprache.
- Jede DKA Sprache ist eine DTM-Sprache.
- Jede NKA Sprache ist eine NTM-Sprache.

BILD

3.11 Fragen

1. Welche Inklusionen sind strikt?
2. Gegeben eine NKA:
   Gibt es einen DEA der genau die gleiche Sprache akzeptiert?

3. Gegeben zwei Maschinen:
   akzeptieren sie die gleiche Sprache?

4. ...

4 (Baby) Mengenlehre

$f : A \to B$ Funktion

Injektion:	$\forall a, b \in A : \quad a \neq b \quad \Rightarrow \quad f(a) \neq f(b)$	
Surjektion:	$\forall b \in B \exists a \in A : f(a) = b$	
Bijektion:	Surjektion & Injektion	(eindeutige Zuordnung)

4.1 Definition

A und B heißen gleich mächtig, wenn es eine Bijektion zwischen A und B gibt.

4.1.1 Beispiel

$\mathbb{G}$ seien die nichtnegativen geraden Zahlen: $\{0, 2, 4, \ldots\}$
$\mathbb{N}$ und $\mathbb{G}$ sind gleichmächtig.

4.1.2 Beweis

$$f(x) = 2 \cdot x \qquad \text{Bijektion!}$$

4.2 Definition

A heißt endlich, wenn $\exists k \in \mathbb{N}$, sodass A gleichmächtig wie $\{0, \ldots, k-1\}$

$$A = \{a_0, a_1, \ldots, a_{k-1}\}$$

A heißt abzählbar, wenn A endlich, oder A gleich mächtig wie $\mathbb{N}$ - abzählbar unendlich.

4.3 Lemma

$A_1, A_2, \ldots A_k$ endliche Mengen, $k \in \mathbb{N}$

$A_1 \cup A_2 \cup \ldots \cup A_k$	ist endlich
$A_1 \times A_2 \times \ldots \times A_k$	ist endlich
A_1^k	ist endlich

4.4 Lemma

$A_0, A_1, \ldots$ sei unendliche Folge von disjunkten endlichen Mengen
Dann ist $\bigcup_{i \in \mathbb{N}} A_i$ abzählbar (undendlich)

4.5 Lemma

A, B abzählbar unendlich $\quad \Rightarrow \quad$ $A \times B$ abzählbar unendlich

4.5.1 Beweis

	0	1	2	3	4
0	$(0,0)$	$(0,1)$	$(0,2)$	$(0,3)$	$(0,4)$
1	$(1,0)$	$(1,1)$	$(1,2)$	$(1,3)$	$(1,4)$
2	$(2,0)$	$(2,1)$	$(2,2)$	$\ldots$	
3	$(3,0)$	$(3,1)$	$\ldots$		

$$C_i = \{(a_j, a_k) \mid j + k = i\}$$
$$A \times B = \bigcup_{i \in \mathbb{N}} C_i$$

C_i ist disjunkt und $|C_i| = i + 1$

4.6 Lemma

Sei A abzählbar unendlich.

$$2^A = \{B | B \subset A\}$$

2^A ist NICHT abzählbar

4.6.1 Beweis

$A = \{a_o, a_1, \ldots\}$ $B \subset A$ kann mit „undendlicher“ Bitfolge s_B identifiziert werden.

$$s_B = \begin{cases} 1 & a_i \in B \\ 0 & a_i \notin B \end{cases}$$

$B = \{1, 3, 7\}$ $\quad$ $010100010\ldots$
$B = \{1, 3, 5, 7, \ldots\}$ $\quad$ $010101010101\ldots$

2^A nicht abzählbar, d.i. $f : \mathbb{N} \to 2^A$
$\Rightarrow$ f ist keine Bijektion (z.B. weil f keine Surjektion)

$f(0)$	$=$	B_0	s_{B_0}	0	1	1	0	0	1	1	$\ldots$	
$f(1)$	$=$	B_1	s_{B_1}	1	1	1	0	0	1	1	$\ldots$	1
$f(2)$	$=$	B_2	s_{B_2}	1	1	1	0	0	1	1	$\ldots$	0

4.7 Cantor'sches Diagonalisierungsverfahren

Möchte zeigen:
$\exists B \subset A$ sodass $\forall n \in \mathbb{N} : f(n) \neq B$
$\Leftrightarrow$
Suche Bitfolge, die in Auflistung nicht vorkommt, sich also von jedem s_{B_i} unterscheidet.

4.8 Satz

A Menge $\Rightarrow$ $\exists$ keine Surjektion $h : A \to 2^A$

4.8.1 Konzept

A Menge $\Rightarrow$ $\exists$ keine Bijektion $h : A \to 2^A$

4.8.2 Beweis

Wollen zeigen:

$$\forall h : A \to 2^A \quad \exists X \in 2^A : \forall a \in A : h(a) \neq X$$
$$X \subseteq A$$

h vorgegeben.

$$\Delta = \{x \in A \mid x \notin h(x)\}$$

Behauptung:

$$\forall a \in A : \ \Delta \neq h(a) \qquad \text{(sie unterscheiden sich in element a)}$$
$$a \in \Delta \Leftrightarrow a \notin h(a)$$

4.9 Satz

$$\underbrace{\text{NTM-Sprachen über } \Sigma}_{L^{\in}} \quad \neq \quad \underbrace{\text{alle Sprachen über } \Sigma}_{2^{\Sigma *}}$$

4.9.1 Beweis

$\Rightarrow$ $\exists$ NTM die genau L akzeptiert.
Es reicht zu zeigen, dass es nur abzählbar viele NTMs gibt.
Σ^* abzählbar unendlich
2^{Σ^*} überabzählbar

Jede NTM als endlichen String über $\Sigma \vee ASCII$ spezifizieren.
Abbildung $(\Sigma \vee ASCII)^*$ ist abzählbar.

5 Endliche Automaten (formale Spezifikation)

$$M = (\Sigma, Q, s, F, \Delta)$$

$Sigma$	Eingabealphabet (endlich)
Q	Zustandsmenge (endlich)
$s \in Q$	Startzustand
$F \subset Q$	Endzustände
$\Delta \subset Q \times \Sigma \times Q$	„Regeln“

Definition

M akzeptiert $w = w_1 w_2 \dots w_n \in \Sigma^*$ genau dann wenn $\exists$ Zustandsfolge

$$q_0 q_1 \dots q_n \text{ mit } \begin{array}{l} q_0 = s \\ q_n \in F \\ \text{für } 0 < i \leq n \, (2_{i-1} w_i, q_i) \in \Delta \end{array}$$

$$L(M) = \{w \in \Sigma^* \mid \text{w wird von M akzeptiert}\}$$

5.1 Beispiel

$$\Sigma = \{a, b\} \quad s = 0 \quad \Delta = \{(0, a, 0), (0, b, 0), (0, a, 1), (1, a, 2)\}$$
$$Q = \{0, 1, 2\} \quad F = \{2\}$$

babaa wird von M akzeptiert

w_1	w_2	w_3	w_4	w_5	
0	0	0	0	1	2

aab wird nicht von M akzeptiert.

5.2 Übergangsgraph eines endlichen Automaten M (Transitionsgraph)

Knotenmenge	Q	
gerichtete Kanten mit Beschriftung	$[q, q'\rangle$	mit Beschriftung a ist im Graphen
	$\Leftrightarrow$	$(q, a, q') \in \Delta$

ZEICHNUNG CHRISTIAN HOLLER 666

$w = w_1 w_2 \dots w_n$ wird von M akzeptiert

$\Leftrightarrow$ $\exists$ gerichteter Pfad P von s zu einem Knoten $q_n \in F$, so dass die Kantenbeschriftungen entlang P genau $w = w_1 w_2 \dots w_n$ ergeben.

5.3 Definition

M heißt deterministisch, wenn

$$\forall (q,a) \in Q \times \Sigma i \,|||q' \;\mid\; (q,a,q') \in \Delta\}| \leq 1$$

sonst heißt M nicht-deterministisch.

5.4 Alternative Definition von L(M)

für jedes $q \in Q$ definiert $\delta_q : \; \Sigma^* \to 2^Q$

$$\delta_q(\epsilon) = \{q\}$$
$$\delta_q(ua) = \bigcup_{q' \in \delta_q(u)} \delta_{q'}(a)$$

$\delta_q(w) =$ „Zustände die von q beginnend über w erreichbar sind“

$$L(M) = \{w \; in \Sigma^* \;\mid\; \delta_s(w) \cap F \neq \emptyset\}$$

5.5 Gibt es Sprachen, die nicht von NEA's akzeptiert werden?

Behauptung

$$L = \{a^n \$ b^n \$ \;\mid\; n \in \mathbb{N}\} \quad \text{wird von keinem NEA akzeptiert.}$$

Beweis

Wollen zeigen $\forall$ NEA's M: $L(M) \neq L$

Sei M NEA, Zustandsmenge Q, $|Q| = m$
Betrachte $a^m \$ b^m \$ \; \in L$

Fall 1

$$a^m \$ b^m \$ \notin L(M) \quad \Rightarrow L(M) \neq L$$

Fall 2

$$a^m \$ b^m \$ \notin L(M) \quad \Rightarrow \exists \underbrace{q_o q_1 \ldots q_m} p_0 p_1 \ldots p_m \quad p \in F$$

Folge von $m+1$ Zuständen
$\Rightarrow$ irgenein Zustand wird wiederholt.

ZEICHNUNG CHRISTIAN HOLLER 667

Korrektur

„NEA-Sprachen $\overset{\subset}{\neq}$ NKA-Sprachen “
„DEA-Sprachen $\overset{\subset}{\neq}$ DKA-Sprachen “

5.6 Pumping Lemma

Sei L eine DEA/NEA Sprache
$\Rightarrow$ $\exists m \in \mathbb{N} : \forall x \in L$ mit $|x| > m$
$\exists$ Unterteilung $x = uvw$ mit $|uv| \leq m$ und $0 < |v| \leq m$
$\forall i \in \mathbb{N}:$ $uv^i w \in L$

Beweis

Sei L EA Sprachen
Sei M eine DEA/NEA mit $L(M) = L$
Q sei Zustandsmenge von M
$m = |Q|$
Sei $\mathcal{G}$ Übergangsgraph von M
s Startzustand
$\mathcal{G}$ hat m Knoten
Sei $x \in L,\ |x| > m$
(wenn nicht existent, muss nichts gezeigt werden!)
Sei P der „akzeptierende Pfad“ für x durch $\mathcal{G}$
Im Anfangsteil der Lnänge m von P muss es Knotenwiederholungen geben.

ZEICHNUNG CHRISTIAN HOLLER 668

$$P \Rightarrow Q \quad \Leftrightarrow \quad \neq Q \Rightarrow \neq P$$

$$\forall m \in \mathbb{N}\ \exists x \in L :\ |x| > m\ \forall \text{ Unterteilungen } \begin{matrix} x = uvw \\ |uv| \leq m \\ 1 \leq |v| \leq m \end{matrix} \quad \forall i \in \mathbb{N}\ uv^i w \notin L$$

Verwendung, um zu zeigen, das $\underline{L}$ nicht NEA-Sprache

1. Gegner gibt m vor
2. „Wir“ geben $x \in L$ an, mit $|x| > m$
3. Gegner gibt Unterteilung $x = uvw$ vor, $|uv| \leq m$, $1 \leq |v| \leq m$
4. „Wir“ geben $i \in \mathbb{N}$ an, sodass $uv^i w \notin L$.

Wenn „Wir“ dieses SPiel immer gewinnen können, dann ist L KEINE NEA-Sprache.

Beispiel

Zeige, dass $L = \{a^n \mid n \text{ ist Primzahl}\}$ keine NEA-Sprache ist.

1. Gegner gibt m
2. Wir wählen eine Primzahl $p > m$ und geben $x = q^p \in L$
3. Gegner: $x = uvw \quad a^p = a^j a^k a^l \quad j + k \leq m,\ 1 \leq k \leq m \quad j + k + l = p$
4. Wir wählen i so, dass $a^j \left(a^k\right)^i a^l \notin L$
   d.i. $j + k + l$ soll keine Primzahl sein
   z.B. $i = j + l \quad j + k(j + l) + l = (k + 1)(j + l)$

$$\delta_q(w) = \{q' \in Q \mid q' \text{ von } q \text{ über Pfad mit Beschreibung w erreichbar}\}$$

$$L_q = \{w\ in\Sigma^* \mid \delta_q(w) \cap F \neq \emptyset\}$$

$$L_\delta = L(m)$$

5.7 Definition

$L \subset \Sigma^*$ irgendeine Sprache

$$C_L :\ \Sigma^* \to 2^{\Sigma^*} \quad C_L(w) = \{x \in \Sigma^* \mid wx \in L\}$$

Forsetzungssprache von w bezüglich L.

Beispiel 1

$$\begin{aligned} L_1 &= \{a^n €b^n € \mid n \in \mathbb{N}\} \\ C_L(aa€b) &= \{b€\} \\ C_L(aa€ab) &= \{\ \} \\ C_L(aa) &= \{€bb€,\ a€bbb€,\ \ldots\} \\ &= \left\{a^n €b^{n+2} € \mid n \in \mathbb{N}\right\} \\ C_{L_1}(a^n €) &= \{b^n €\} \end{aligned}$$

Beispiel 2

$$L = \{w \ in \{a,b\}^* \ \mid \ (\#a' smw) \ mod \ 3 = 0\}$$

$$\begin{aligned} C_L(aaa) &= L \\ C_L(ababbaaaaa) &= L \\ C_L(baa) &= \{x \ in \{a,b\}^* \ \mid \ (\#a' smx) \ mod \ 3 = 1\} = L_1 \\ C_L(bab) &= \{x \in \{a,b\}^* \ \mid \ (\#a' smx) \ mod \ 3 = 2\} = L_2 \\ C_L(abbaaab) &= L_2 \end{aligned}$$

5.8 Satz (Myhill-Nerode)

L ist DEA-Sprache $\Leftrightarrow \mathcal{F}_L = \{C_L(w) \ \mid \ w \in \Sigma^*\}$ ist endlich.

Beweis

Teil (i)i „⇒“

$$L \text{ ist DEA-Sprache} \Rightarrow \ \exists \text{ DEA M} \ \begin{array}{l} \Sigma \\ Q \\ s \\ F \\ \Delta \text{ deterministisch} \end{array} \quad \text{so dass } L = L(M).$$

$$w \in \Sigma^* : \ C_L(w) = \begin{cases} L_p & \text{falls } \delta_q(w) = \{p\} \\ \emptyset & \text{sonst} \end{cases}$$

$$|Q| = n \quad \mathcal{F}_L = \{L_p \ \mid \ p \in Q\}$$

(unter der Annahme, dass jeder Zustand $q \in Q$ von s erreichbar ist)

$$|\mathcal{F}| \leq |Q| + 1 \quad \text{endlich}$$

Teil (ii) „⇐“

$$L \subset \Sigma^*, \quad \mathcal{F}_L \text{ endlich}$$

Wir wollen einen DEA bauen, der genau L akzeptiert:

$$\begin{aligned} &\Sigma \\ &Q = \mathcal{F}_L \\ &s = C_L(\epsilon) = L \\ &F = \{L' \in \mathcal{F}_L \ \mid \ \epsilon \in L'\} \\ &\quad = \{C_L(w) \ \mid \ w \in L\} \\ &\Delta = \{(C_L(w)), a, C_L(wa) \ \mid \ w \in \Sigma^*, a \in \Sigma\} \end{aligned}$$

Behauptung 1

M ist deterministisch

Behauptung 2

M akzeptiert genau L

Beweis zu Behauptung 1

müssen zeigen:

$$v, w \in \Sigma^*, a \in \Sigma \quad C_L(v) = C_L(w) \Rightarrow C_L(va) = C_L(wa)$$

$$\underbrace{x \in C_L(va)}_{\Leftrightarrow vax \in L} \Leftrightarrow \underbrace{ax \in C_L(v)}_{=C_L(w)} \Leftrightarrow wax \in L \Leftrightarrow x \in C_L(wa)$$

Beweis zu Behauptung 2

$$w = w_1 w_2 \dots w_n \in L$$

Anwendung von Satz (Myhill-Nerode)

$$\begin{array}{c} w_1 w_2 \dots w_n \in C_L(\epsilon) \\ \Updownarrow \\ w_2 \dots w_n \in C_L(\epsilon) \\ \Updownarrow \\ w_3 \dots w_n \in C_L(\epsilon) \\ \vdots \\ w_n \in C_L(w_1 \dots w_{n-1}) \\ \Updownarrow \\ \epsilon \in C_L(w_1 \dots w_n) \end{array}$$

$$\begin{array}{c} w_1 w_2 \dots w_n \notin C_L(\epsilon) \\ \Downarrow \\ w_2 \dots w_n \notin C_L(w_1) \\ \vdots \\ \epsilon \notin C_L(w_1 w_2 \dots w_n) \quad \text{kein Endzustand} \end{array}$$

Anmerkung

5.9 Satz

L ist NEA-Sprache $\Rightarrow$ L ist auch eine DEA-Sprache

5.9.1 Beweis

L ist NEA-Sprache

d.i. $\exists$ nicht deterministischer Automat $M \begin{matrix} \Sigma \\ Q \\ s \\ F \\ \Delta \end{matrix}$ $L(M) = L$

Nach Myhill-Nerode reicht es zu zeigen, dass $\mathcal{F}_L$ endlich.

$$w \in \Sigma^* : \; C_L(w) = \bigcup \{L_{q'} \;\mid\; q' \in \delta_q(w)\}$$

Q ist endlich $\Rightarrow$ # der verschiedenen $\delta_q(w)$'s endlich.

$$\left\{\delta_q(w) \;\mid\; |w \in \Sigma^*| \leq 2^{|Q|}\right\}$$

$$\Rightarrow \{C_L(w) \;\mid\; w \in \Sigma^*\} \quad \text{endlich}$$

Direkte Konstruktion

Aus NEA M konstruiere DEA M', so dass $L(M') = L(M)$

$$\begin{matrix} & \Sigma & & \Sigma & & \\ & Q & & Q' & = & 2^Q \\ M & s \in Q & M' & s' & = & \{s\} \\ & F \subset Q & & F' & = & \{A \subset B \;\mid\; A \cap F \not\approx Q\} \\ & \Delta & & \Delta' & = & \left\{(A, a, B) \;\mid\; A \subseteq Q, B = \bigcup_{\xi \in A} \delta_q(a)\right\} \end{matrix}$$

Idee:

$\delta_q(w) \ldots$ Zustände von M'

Beispiel

CHRISTIAN HOLLER ZEICHNUNG 1337

5.10 „ϵ-Übergänge “

Idee

endlicher Automat muss Lesekopf nicht bei jedem Übergang nach rechts rücken, Kopf darf auch verweilen.

$$\Sigma$$
$$Q$$
$$s \in Q, F \subset Q$$
$$\Delta \subseteq Q \times \Sigma^{\leq 1} = \Sigma^1 \cup \{\epsilon\}$$

M akzeptiert $x \in \Sigma^*$

$$\exists q_0, \ldots, q_m \in Q,\ \exists x_1, x_2, \ldots, x_m \in \Sigma^{\leq 1}$$

für $1 <\leq i \leq m : (q_{i-1}, x_i, q_i) \in \Delta$

Übergangsgraphen

Kanten können auch mit ϵ beschriftet werden.

ZEICHNUNG CHRISTIAN HOLLER 1338

5.11 Satz

Wenn L durch einen NEA mit ϵ-Übergängen akzeptiert wird, dann ist L regulär.
d.i. L wird auch durch DEA (oder NEA) ohne ϵ-Übergange akzeptiert.

Beweis

Methode 1

Zeige $\mathcal{F}_L$ ist endlich

Methode 2

Konstruiere aus NEA mit ϵ-Übergängen M, einen NEA ohne ϵ-Übergänge M', so dass $L(M) = L(M')$.

Es reicht zu zeigen, wie man eine ϵ-Kante entfernen kann!

ZEICHNUNG CHRISTIAN HOLLER 1339

6 2-Wege Automat

Idee

deterministischer endlicher Automat
Lesekopf darf auch zurückrücken.

Übergangsregeln

$$(q, a, \epsilon', \nu) \in Q \times \Sigma \times Q \times \{L, B, R\}$$

6.1 Satz

Wenn L durch einen deterministischen 2-Wege-Automaten akzeptiert wird, dann ist L regulär.

Beweis

Idee

Zeige, L hat endlich viele Forsetzungssprachen.

$$ZEICHNUNG\ CHRISTIAN\ HOLLER\ 1337.kA$$

$$f_x : \ Q \to Q \cup \{\text{term, div}\}$$

Nehmen an, Maschine ist im Zustand ϵ und hat Lesekopf auf erstem Zeichen von x und Maschine läuft.

1. Maschine rückt Lesekopf irgendwann über linkes Ende von x und ist dann im Zustand ϵ'
$$f_x(\epsilon) = \epsilon'$$
2. Maschine terminiert ohne über linkes Ende von x zu rücken.
$$f_x(\epsilon) = \text{term}$$
3. sonst
$$f_x(\epsilon) = \text{div}$$

$$x, y \in \Sigma^*$$

$$f_x = f_g \quad \Rightarrow \quad \forall m \in \Sigma^* : \quad wx \in L \quad \Leftrightarrow \quad wy \in L$$
$$L_f = \{x \in \Sigma^* \ \mid \ f_x = f\}$$

$$w \in \Sigma^* \quad F_w = \{f_x \ \mid \ wx \in L \text{ von M akzeptiert}\}$$
$$C_L(w) = \bigcup \{L_f \ \mid \ f \in F_w\}$$

$$F = \{f : Q \to Q \cup \{\text{div, term}\}\} \quad \text{endlich}$$

$\Rightarrow$ es gibt nur endlich viele $C_L(u)$'s

7 Abschlusseigenschaften der regulären Sprachen

7.1 Satz

Es seien L und L' reguläre Sprachen über Σ. Darum sind auch die folgenden Sprachen regulär:

1. $\bar{L} = \{w \in \Sigma^* \mid w \notin L\}$
2. $L\ L' = \{uv \mid u \in L,\ v \in L'\}$
3. $L \cup L'$
4. $L \cap L'$
5. $L^* = \{w_1 w_2 \dots w_k \mid k \in \mathbb{N}, w_i \in L,\ 1 \leq i \leq k\}$

Beweis

$$L \text{ regulär } \rightarrow \exists DEA\ M\ textmit\ L(M) = L$$
$$L' \text{ regulär } \rightarrow \exists DEA\ M'\ textmit\ L(M') = L'$$

o.B.d.A. nehmen wir an, M und M' seien vollständig spezifiziert.

1. $\forall q \in Q,\ \forall a \in \Sigma,\ \exists_1 q' \in Q,\ (q, a, q') \in \Delta$
2. alle $q \in Q$ vom Startzustand erreichbar.

Wenn nicht vollständig spezifiziert:

1. $\tilde{Q} = Q \cup \{\perp\}$
   Es gibt keinen Übergang, der von q mit a wegführt
   $\rightarrow$ erzeuge zusätzlichen Übergang $(q, a, \perp)$
2. entferne nicht von s erreichbare Zustände

1. Maschine für $\bar{L}$

   Σ
   Q
   s
   $F = Q \setminus F$
   Δ

2. $ZEICHNUNG\ SEBASTIAN\ WAINER$ 5.11.2004.1

   Σ
   $Q \cup Q'$
   s
   F'
   $\Delta \cup \Delta' \cup \{(q, \epsilon, s') \mid q \in F\}$

3.

ZEICHNUNG SEBASTIAN WAINER 5.11.2004.2

Σ
$Q \cup Q'$
$\tilde{s}$
$F \cup F'$
$\Delta \cup \Delta' \cup \{(\tilde{s}, \epsilon, s'), (\tilde{s}, \epsilon, s)\}$

4. $A \bar{\cup} B = \bar{A} \cup \bar{B}$
$L \cap L' = \overline{\bar{L} \cup \bar{L}'}$

Lasse M und M' parallel laufen.

Σ
$Q \times Q'$
(s, s')
$\tilde{F} = \{(q, q') \mid q \in F,\ q' \in F'\}$
$\tilde{\Delta} = \{(p, p'), a, (q, q') \mid (p, a, q) \in \Delta,\ (p', a, q') \in \Delta'\}$

5.

ZEICHNUNG SEBASTIAN WAINER 5.11.2004.3

8 Entscheidungseigenschaften von regulären Sprachen

8.1 Satz:

„Mann kann entscheiden, ob für eine gegebene reguläre Sprache $L \subset \Sigma^*$ gilt $L = \Sigma^*$.“
Gegeben DEA M, kann entschieden werden, ob $L(M) = \Sigma^*$.

8.1.1 Beweis

o.B.d.A M vollständig spezifiziert.
M akzeptiert $\Sigma^* \iff F = Q$

8.2 Satz

„Gegeben DIA M, kann entschieden werden, ob $L(M) = \{\}$, M akzeptiert $\{\}$ $\iff$ alle von s erreichbaren Zustände sind nicht in F“

Frage

M, M' $DEA's$
kann man entscheiden, aber $L(M) = L(M')$?

Antwort

JA! denn $L, L' \subseteq \Sigma^*$

$$L = L' \iff L \cup \bar{L'} = \Sigma^* \text{ und } L \cap \bar{L'} = \{\}$$

wegen Abschlußeigenschaften sind $L \cup \bar{L'}$ und $L \cap \bar{L'}$ regulär ...

9 „Minimierung“ von DEA'n

Problem

Gegeben: DEA $\tilde{M'}$ mit möglichst wenigen Zuständen, sodass $L(M) = L(\tilde{M})$.

$$\begin{aligned} \delta_q(w) &= \{q' \mid q' \text{ von } q \text{ über Pfad mit Beschreibung } w \text{ erreichbar}\} \\ L_q &= \{w \in \Sigma^* \mid \delta_q(w) \subseteq F\} \end{aligned}$$

Falls $q \neq q'$ gilt:
$L_q = L_{q'}$ heißen q, q' ununterscheidbar und man kann auf einen der beiden verzichten.

Falls $L_q \neq L_{q'}$ dann heißen q und q' unterscheidbar.

Beachte: Unterscheidbarkeit ist eine Äquivalenzrelation auf Q

$$\begin{gathered} q \sim q' \Rightarrow q' \sim q \\ q \sim q \\ q \sim q' \wedge q' \sim q'' \Rightarrow q \sim q' \end{gathered}$$

d.i. Äquivalenzklassen können zusammengefasst werden, und bilden Zustände des minimalen Automaten.

Wie bestimmt man alle unterscheidbaren Zustandspaare U ?

Regel 1

$$q \in F,\ q' \notin F \Rightarrow \{q, q'\} \in U$$

Regel 2

$$q, q' \in Q, a \in \Sigma : \begin{matrix} \{p\} = \delta(a) \\ \{p'\} = \delta_{q'}(a) \end{matrix} \{p, p'\} \in U \Rightarrow \{q, q'\} \in U$$

„Table-filling“ Algorithmus (zur Bestimmung von U)

1. Wende Regel 1 an, solange möglich
2. Wende Regel 2 an, solange möglich

Hinweis vom Autor

Es werden die Zustände verglichen die von dem jeweiligen Punkt erreichbar sind.

Behauptung

Wenn der Algorithmus terminiert, umfasst U genau alle unterscheidbaren Paare von Zuständen. Die restlichen Paare sind ununterscheidbar.

Beispiel

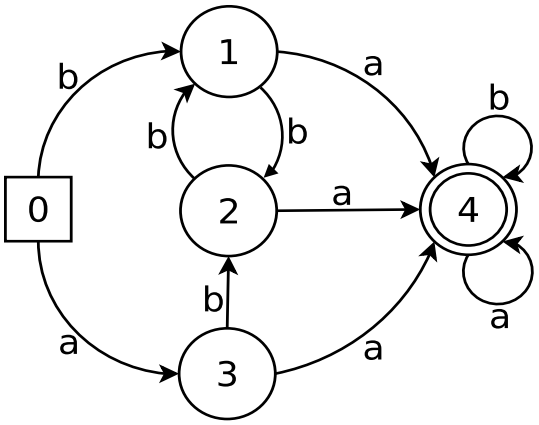
ZEICHNUNG SEBASTIAN WAINER 5.11.2004.4

	3	2	1	0
0	×	×		
1	×	×		
2	×			
3				

$$\{0, 1\} \notin U$$

d.i. Zustände 0 und 1 sind ununterscheidbar und bilden eine „Äquivalenzklasse“ (Seite 28)

Beispiel: Keller-Automat



$$U = \{\{2,4\} \mid 0 \leq i \leq 3\}$$

0	■	■	■	■	■
1	×	■	■	■	■
2	×		■	■	■
3	×			■	■
4	×	×	×	×	■
	0	1	2	3	4

$$\{0,1\} \xrightarrow{a} \{3,4\} \in U$$
$$\Rightarrow \{0,1\} \in U$$

$$\{0,2\} \xrightarrow{a} \{3,4\} \in U$$
$$\Rightarrow \{0,2\} \in U$$

$$\{0,3\} \xrightarrow{a} \{3,5\} \in U$$
$$\Rightarrow \{0,3\} \in U$$

$$\{1,2\} \xrightarrow{a} \{4,4\} \notin U$$
$$\{1,2\} \xrightarrow{b} \{1,2\} \notin U$$

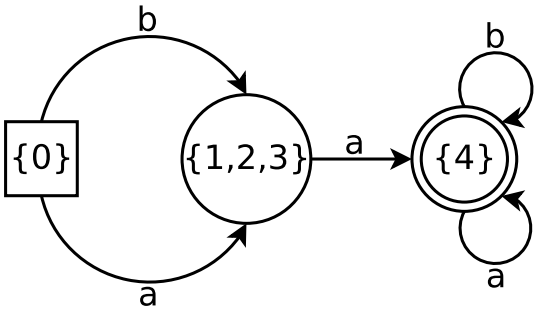
$$\{1,3\} \xrightarrow{a} \{4,4\} \notin U$$
$$\{1,3\} \xrightarrow{b} \{2,2\} \notin U$$

$$\{2,3\} \xrightarrow{a} \{4,4\} \notin U$$
$$\{2,3\} \xrightarrow{b} \{1,2\} \notin U$$

$$2 \quad \equiv \quad 3 \quad \equiv \quad 1$$

Äquivalenzklassen

$$\{0\}, \qquad \{1,2,3\}, \qquad \{4\}$$



9.1 Korrektheit

9.1.1 Lemma 0

$\forall\ \{p, q\} \in U$ gilt: p und q sind unterscheidbar; **d.i.**

$$L_p \neq L_q$$

Beweis

Stimmt nach Schritt 2, denn:

$$p \in F \Leftrightarrow \epsilon \in L_p$$
$$q \notin F \Leftrightarrow \epsilon \notin L_q$$

Regelanwendung in Schritt 3:

$$\{q, q'\} \in U \rightarrow L_q \neq L_{q'} \rightarrow \exists w \in L_q \vee w \notin L_{q'}$$

Regel

$\Rightarrow\ aw \in L_p \vee aw \notin L_{p'} \Rightarrow L_p \neq L_{p'}$
$\Rightarrow \{p, p'\} \in U$

9.1.2 Lemma 1

Sobald in Schritt 3 Reegel nicht mehr anwendbar ist, gilt für jedes Paar $p \neq q$ und $\{p, q\} \notin U$, dass $L_p = L_q$, also p und q sind NICHT unterscheidbar.

Beweis

Sei (mit U nach Terminierung des Algorithmus)

$$X = \{\{p, q\} \notin U \ \mid\ L_p \neq L_q\}$$

$X = \emptyset$ stimmt Lemma

$X \neq \emptyset$ Sei $w \in \Sigma^*$ das kürzeste Wort, das für irgenein $\{p, q\} \in X$ folgendes erfüllt:
$w \in L_p,\ w \notin L_q$

$$p \overset{w_1}{\curvearrowright} p_1 \overset{w_2}{\curvearrowright} p_2 \overset{w_3}{\curvearrowright} p_3 \curvearrowright \ldots \overset{m}{\curvearrowright} p_m \quad \in F$$

$$q \overset{w_1}{\curvearrowright} q_1 \overset{w_2}{\curvearrowright} q_2 \overset{w_3}{\curvearrowright} q_3 \curvearrowright \ldots \overset{m}{\curvearrowright} q_m \quad \notin F$$

$p_1,\ q_1$ unterscheidbar!

Fall 1

$$\{p_1, q_1\} \in U$$

Algorithmus darf noch nicht terminiert haben: Regel im Schritt 3 noch anwendbar!

Fall 2

$$\{p_1, q_1\} \notin U \rightarrow \{p_1, q_1\} \in X\ (L_{p_1} \neq L_{q_1})$$

kürzer als w im Widerspruch zu Wahl von v.

10 Reguläre Ausrücke [ρ] Beschreibungsart für Sprachen

ρ	regulärer Ausdruck	$\underbrace{L(\rho)}_{\subseteq \Sigma^*}$
$\emptyset$	regulärer Ausdruck	$L(\emptyset) = \{\}$
ϵ	regulärer Ausdruck	$L(\epsilon) = \{\epsilon\}$
a	regulärer Ausdruck	$L(a) = \{a\} \quad \forall a \in \Sigma^*$
α, β	reguläre Ausdrücke	$\Rightarrow \alpha\beta,\ (\alpha + \beta),\ (\alpha)^*$ sind reguläre Ausdrücke

$$L(\alpha\beta) = L(\alpha)L(\beta) = \{uv \mid u \in L(\alpha),\ v \in L(\beta)\}$$
$$L(\alpha + \beta) = L(\alpha) \cup L(\beta)$$
$$L\left((\alpha)^*\right) = L((a))^*$$

10.1 Satz

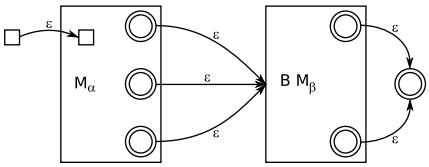
	$L = L(\rho)$	für irgendeinen regulären Ausdruck ρ
$\Leftrightarrow$	$L = L(M)$	für irgendeinen endlichen Automaten M

Beweis

„$\Rightarrow$“

Gegeben: ρ, wohldefinierter endlicher Automat M_{rho}, mit $L(\rho) = L(M_\rho)$

$\emptyset$	$M_\emptyset$	$\square$
ϵ	M_ϵ	$\square$
a	M_a	$\square \overset{a}{\curvearrowright} \square$



„⇐“

Sei M ein deterministischer Endlicher Automat

$$Q = \{1, \ldots, n\} \qquad \Sigma$$
$$\rho = 1$$

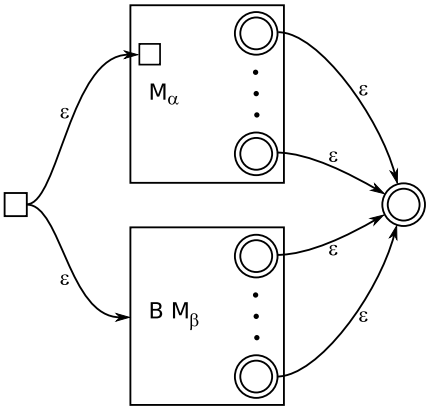
G sei Übergangsgraph von M, F Endzustände

$R_{ij}^k \subset \Sigma^*$ Menge aller Beschriftungen von Pfaden in G mit Anfangszuständen i, Endknoten j und inneren Knoten $\leq k$.
$i \curvearrowright k_1 \curvearrowright \ldots \curvearrowright j$

$$L(M) = \bigcup_{q \in F} R_{iq}^k$$

Also wenn q_{ij}^k regulärer Ausdruck mit $L\left(q_{ij}^k\right) = R_{ij}^k$ dann

$$L(M) = L\left((\rho_{iq_1}^n + \rho_{iq_2}^n) + \ldots \rho_{iq_e}^n\right) \qquad F = \{q_1, \ldots, q_e\}$$
$$R_{ii}^0 = \{\epsilon\} \cup \bigcup \{\{a\}^* \mid (i, a, i) \in \Delta\}$$
$$R_{ij}^0 = \{a \in \Sigma \mid (i, a, j) \in \Delta\}$$



Annahme

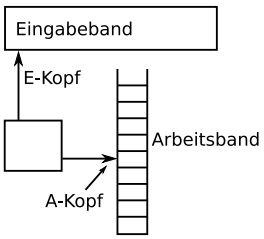
R_{ij}^k bekannt für alle i, j

$$R_{ij}^{k+1} = R_{ij}^k \quad \cup \quad R_{i(k+1)}^k \left(R_{(k+1)(k+1)}^k\right)^* R_{i(k+1)}^k$$

$$i \overset{\rightsquigarrow}{\leq} k\ k+1 \overset{\rightsquigarrow}{\leq} k\ k+1 \overset{\rightsquigarrow}{\leq} k\ k+1 \overset{\rightsquigarrow}{\leq} k\ j$$

$$\rho_{ij}^{k+1} = \rho_{ij}^{k} + \rho_{i(k+1)}^{k} \left(\rho_{(k+1)(k+1)}^{k}\right)^{*} \rho_{i(k+1)}^{k}$$

11 Kellerautomaten



Eingabekopf	bleiben
	nach rechts rücken (nichts schreiben!!)
Arbeitskopf	bleiben
	nach unten rücken (löscht Zeichen)
	nach oben rücken und schreiben

11.1 Regeln: je nach Zustand

			neuer Zustand
Eingabekopf	*symbol*	$\longrightarrow$	Eingabekopf Bewegung
Arbeitskopf	*symbol*	$\longrightarrow$	Arbeitskopf Bewegung

11.2 Formalisierung

NKA	*Nichtdeterministischer Kellerautomat M*
Σ	Eingabealphabet
Γ	Arbeitsalphabet
Q	Zustandsmenge (endlich)
$s \in Q$	Startzustände
$F \subseteq Q$	Endzustände
$€ \in \Gamma$	Unterstes Symbol auf Arbeitsband

$$\Delta \subseteq \left(Q \times \underbrace{\Sigma^{\leq 1} \times}_{\Sigma^0 \cup \Sigma^1} \Gamma \right) \times \Gamma^* \times Q \qquad \text{Regeln}$$

11.3 informelle Erklärung

$$(q, a, A, U, q') \in \Delta$$

Wenn in Zustand q

Eingabekopf liest a
Arbeitskopf liest A

neuer Zustand q'

Eingabekopf rückt nach rechts
Arbeitskopf ersetzt A durch U
$U = \epsilon \rightarrow$ Arbeitskopf rückt nach rechts unten
$U = B \rightarrow$ Arbeitskopf ersetzt A durch B_k und bleibt
$U = B_1 B_2 \dots B_k \rightarrow$ Arbeitskopf ersetzt A durch B_1, rückt nach oben und schreibt $B_{k-1} \dots B_1$

11.4 Übereinkunft

Alphabetsanfangsbuchstaben		
klein	a,b,c,...	stehen für einzelne Symbole in Σ
groß	A,B,C,...	stehen für einzelne Symbole in Γ
Alphabetsendbuchstaben		
klein	z,y,x,...	$\in \Sigma^*$
greoß	Z,Y,X,...	$\in \Gamma^*$

11.5 Welche Worte werden von M akzeptiert?

Konfigurationen von M

$$K_M = Q \times \Sigma^* \times \Gamma^*$$

M in Konfiguration (q, u, W) bedeutet:
M und Zustand q
Arbeitsbandinhalt W
Eingabebandinhalt über und rechts vom Lesekopf ist u

11.6 Relation

$$\overset{M}{\longmapsto} \text{ auf } K_M$$

$$k \overset{M}{\longmapsto} k'$$

soll ausdrücken: aus Kopf k kommt M durch 1 Regelanwendung.

$$(q, ax, BY) \overset{M}{\longmapsto} (q', x, UY) \quad \textbf{g.d.w} \quad (q, a, B, U, q') \in \Delta$$

$$(q, x, BY) \overset{M}{\longmapsto} (q', x, UY) \quad \textbf{g.d.w} \quad (q, \epsilon, B, U, q') \in \Delta$$

Anfangskonfiguraton von M für $x \in \Sigma^*$

$$(s, x, €) = Init_M(x)$$

Endkonfiguration von M

$$Fin_M = \{(q, €, U) \mid q \in F\}$$

M akzeptiert $x \in \Sigma^*$ genau dann wenn:
$\exists\ k_0, k_1, \ldots, k_m \in K_M$
mit $k_0 = Init_M(x)$ und $k_m \in Fin_M$
und $k_i \overset{M}{\longmapsto} k_{i+1}$ für $0 \leq i < m$

$$Init_M(x) \overset{M}{\longmapsto} k_1 \overset{M}{\longmapsto} k_2 \overset{M}{\longmapsto} \ldots \overset{M}{\longmapsto} k_{m-1} \overset{M}{\longmapsto} k_m \in Fin_M$$

Äquivalent dazu

Sei $\overset{*}{\underset{M}{\longmapsto}}$ die reflexive, transitive Hülle von $\overset{M}{\longmapsto}$.

$$M \text{ akzeptiert } x \quad \Leftrightarrow \quad \exists k \in Fin_M :\ Init_M(x) \overset{*}{\underset{M}{\longmapsto}} k$$

$$L(M) = \{x \in \Sigma^* \mid M \text{ akzeptiert } x\}$$

Beispiel

NKA für die Sprache $\left\{ww^R \mid w \in \{a, b\}^*\right\}$

$$\begin{aligned}
&Q = \{s, q, f\} \qquad s = s \qquad F = \{f\} \\
&\Sigma = \{a, b\} \qquad \Gamma = \{€, A, B\} \qquad € \\
&\quad \Delta = \{ \\
&\qquad\quad (s, a, €, A€, s), \\
&\qquad\quad (s, b, €, B€, s), \\
&\qquad\quad (s, a, A, AA, s), \\
&\qquad\quad (s, a, B, AB, s), \\
&\qquad\quad (s, b, A, BA, s), \\
&\qquad\quad (s, b, B, BB, s), \\
&\qquad\quad (s, \epsilon, A, A, q), \\
&\qquad\quad (s, \epsilon, B, B, q), \\
&\qquad\quad (s, \epsilon, €, €, q), \\
&\qquad\quad (q, a, A, \epsilon, q), \\
&\qquad\quad (q, b, B, \epsilon, q), \\
&\qquad\quad (q, \epsilon, €, \epsilon, f) \qquad \}
\end{aligned}$$

Eingabebeispiel 1

$$baab$$

$$\begin{array}{rcl}
(s, baaab, €) & \overset{M}{\longmapsto} & (s, aab, B€) \\
& \overset{M}{\longmapsto} & \textit{Regel } 4 \\
& & (s, ab, AB€) \\
& \overset{M}{\longmapsto} & \textit{Regel } 7 \\
& & (q, ab, AB€) \\
& \overset{M}{\longmapsto} & \textit{Regel } 10 \\
& & (q, b, B€) \\
& \overset{M}{\longmapsto} & \textit{Regel } 11 \\
& & (q, \epsilon, €) \\
& \overset{M}{\longmapsto} & \textit{Regel } 12 \\
& & (f, \epsilon, \epsilon) \\
& & \in Fin_M
\end{array}$$

Eingabebeispiel 2

$$baaba$$

$$\begin{array}{rcl}
(s, baaab, €) & \overset{M}{\longmapsto} & (s, aaba, B€) \\
& \overset{M}{\longmapsto} & \textit{Regel } 4 \\
& & (s, aba, AB€) \\
& \overset{M}{\longmapsto} & \textit{Regel } 7 \\
& & (q, aba, AB€) \\
& \overset{M}{\longmapsto} & \textit{Regel } 10 \\
& & (q, ba, B€) \\
& \overset{M}{\longmapsto} & \textit{Regel } 11 \\
& & (q, a, €) \\
& \overset{M}{\longmapsto} & \textit{Regel } 12 \\
& & (f, a, \epsilon) \\
& & \notin Fin_M
\end{array}$$

(f, a, ϵ) ist kein gültiger Endzustand
Der Automat hat nicht die ganze Eingabe gelesen.

11.7 Definition: Einfacher nichtdeterministischer Kellerautomat (NKA)

alle Regeln haben Form

$$\left(q, \begin{array}{c} a \\ \epsilon \end{array}, B, U, q'\right)$$

mit

$$U = \begin{cases} \epsilon & (POP) \\ A & \text{ersetzen} \\ CB & PUSH\ C \end{cases}$$

11.8 Lemma

Für jeden $NKA\ M$ gib es einen einfachen $NKA\ M'$ mit

$$L(M) = L(M')$$

Beweis

Ersetze jede Regel

$$\left(q, \begin{matrix} a \\ \epsilon \end{matrix}, B, U, q'\right)$$

von M mit $|U| > 1$ durch Regeln

$$\begin{array}{l} \left(q, \begin{matrix} a \\ \epsilon \end{matrix}, B, U_k, q_k\right) \\ (q, \epsilon, U_k, U_{k-1}, q_{k-1}) \\ (q_i, \epsilon, U_k, U_{i-1}, q_{i-1}) \quad \text{für } 1 < i \leq k \end{array}$$

$$q_2, \ldots, q_{k-1}$$

sind neue Zustände nur für die Regel.

Konfiguration

$$\left(\underbrace{Q}_{\text{Zustand}} \times \underbrace{\Sigma^*}_{\text{Eingabewert}} \times \underbrace{\Gamma^*}_{\text{Kellerinhalt}}\right)$$

$$(q, ax, AU) \overset{M}{\longmapsto} (q', ax, BU) \text{falls } (q, \epsilon, A, B, q')$$

$$\overset{M}{\longmapsto}^* \quad \text{reflexive transitive Hülle}$$

$$k \overset{M}{\longmapsto}^* k'$$

$$kw \in L(U) \Longleftrightarrow (s, w, €) \overset{M}{\longmapsto}^* (f, \epsilon, U)$$

12 Pumping Lemma für NKA Sprachen

12.1 Lemma

Jede NKA Sprache hat die folgende „Pumping Eigenschaft“

$$P: \ \exists N \in \mathbb{N} \quad \forall z \in K, \ |z| > N \quad \exists \text{Unterteilung} \quad z = uvwxy$$
$$|vwx| \leq N \qquad |vx| \geq 1$$
$$\forall i \in \mathbb{N}: \quad uv^i wx^i y \in L$$

Um zu zeigen, dass L keine NKA Sprache ist, genügt es zu zeigen, dass L nicht die Eigenschaft P besitzt.
Also für L gilt $\neg P$:

$$\forall N \in \mathbb{N} \quad \exists z \in L, \ |z| > N \quad \forall z = uvwxy \quad \exists z \in \mathbb{N}: \ uv^i wx^i y \notin L$$
$$|vwx| \leq N \qquad |vx| \geq 1$$

Spiel

1. Gegner gibt $N \in \mathbb{N}$ vor
2. Wir produzieren $z \in L, \ |z| > N$
3. Gegner produziert Unterteilung $z = uvwxy$
4. Wir produzieren $i \in \mathbb{N}$, sodass $uv^i wx^i y \notin L$

12.2 Lemma

Die Sprache $L = \{a^n b^n c^n \ \mid \ n \in \mathbb{N}\}$ ist keine NKA Sprache.

Beweis

Zeige L hat nicht Eigenschaft P

1. Gegner produziert N
2. Wir wählen $z = a^N b^N c^N \in L$
3. Gegner produziert Unterteilung $z = uvwxy, \ |vwx| \leq N, \ |vx| \geq 1$
   also vwx enthält keine a's (*Fall 1*)oder keine c's (*Fall 2*).
4. Wir wählen i

   Fall 1

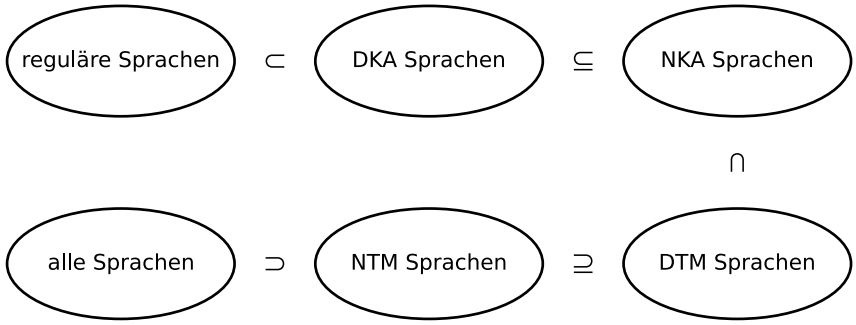
$$\begin{aligned} uv^i wx^i y &= uwy \\ |uwy| &< 3N \quad \text{also } uwy \text{ enthält } N \ a\text{'s} \\ \text{weil } |vx| &\geq 1 \qquad\qquad \text{also } uwy \notin L \end{aligned}$$

Fall 2

analog

12.3 Korrolar

NKA Sprachen sind echte Teilmengen der DTM Sprachen.
(Wenn man glaubt, dass L von einer DTM erkannt werden kann)



12.3.1 Beweis des Pumping Lemmas

L NKA Sprache

$$\text{NKA } M \quad \begin{matrix} \Sigma \\ Q \\ \Gamma \\ q \in Q \\ F \subset Q \\ € \in \Sigma \\ \Delta \end{matrix} \quad \text{mit } M(M) = L$$

o.B.d.A
M einfach, M „vollständig“
$|F| = 1$ und jede akzeptierende Berechnung endet mit leerem Keller.

Statusgraph von M ($\mathcal{S}G(M)$)

Status von M $\quad (q, U) \in Q \times \Sigma^*$

$\mathcal{S}G(M)$ Knotenweg gerichtete Kanten beschriftet mit String aus $\Sigma^{\leq 1}$ $\quad Q \times \Gamma^*$

$$\begin{array}{lcll}
\begin{pmatrix} q, & \begin{matrix} A \\ W \end{matrix} \end{pmatrix} & \xrightarrow{w} & (q', W) & \text{g.d.w.} \quad (q, y, A, \epsilon, q') \in \Delta \\
\begin{pmatrix} q, & \begin{matrix} A \\ W \end{matrix} \end{pmatrix} & \xrightarrow{y} & \begin{pmatrix} q', & \begin{matrix} B \\ W \end{matrix} \end{pmatrix} & \text{g.d.w.} \quad (q, y, A, B, q') \in \Delta \\
\begin{pmatrix} q, & \begin{matrix} A \\ W \end{matrix} \end{pmatrix} & \xrightarrow{y} & \begin{pmatrix} \begin{matrix} B \\ q', \quad A \end{matrix} \\ W \end{pmatrix} & \text{g.d.w.} \quad (q, y, A, BA, q') \in \Delta
\end{array}$$

Wir schreiben

$$(q, W) \overset{u^{\leq \Sigma^*}}{\times \twoheadrightarrow} (q', W')$$

wenn es einen Pfad

$$(q_0, W_0) \overset{u_1}{\to} (q_1, W_1) \overset{u_2}{\to} \ldots \overset{u_m}{\to} (q_m, W_m)$$

in $\mathcal{SG}(M)$ gibt mit

$$\begin{array}{rcl} (q_0, W_0) & = & (q, W) \\ (q_m, W_m) & = & (q', W') \end{array}$$

und $u_1 u_2 \ldots u_m = u$

Beachte

M akzeptiert $u \in \Sigma^*$ genau dann wenn

$$(\underbrace{s, \in\!\!\!\!/}_{\text{Anfangsstatus}}) \times \overset{u}{\twoheadrightarrow} (\underbrace{f, \epsilon}_{\text{Endstatus}})$$

12.4 Definition

Ein Pfad

$$(q_0, W_0) \overset{u_1}{\to} (q_1, W_1) \overset{u_2}{\to} \ldots \overset{u_m}{\to} (q_m, W_m) \quad \in \mathcal{SG}(M)$$

heißt Kellersuffix U erhaltend, wenn fpr $0 \leq i \leq m$ gilt

$$W_i = \begin{matrix} W_i \\ U \end{matrix}$$

$$(q_0, W_0) \overset{\overset{V}{\twoheadrightarrow}}{U} (q_m, W_m)$$

12.5 Hauptbeobachtung

$$\begin{pmatrix} & X \\ q, & A \\ & Y \end{pmatrix} \overset{\overset{v}{\twoheadrightarrow}}{\begin{matrix} A \\ Y \end{matrix}} \begin{pmatrix} & X' \\ q', & A \\ & Y \end{pmatrix} \Longrightarrow \forall W \in \Gamma^* : \begin{pmatrix} & X \\ q, & A \\ & W \end{pmatrix} \overset{\overset{v}{\twoheadrightarrow}}{\begin{matrix} A \\ W \end{matrix}} \begin{pmatrix} & X' \\ q', & A \\ & W \end{pmatrix}$$

Beweisidee fürs Pumping Lemma

Numm an fur $z \in L$ gibt es akzeptierten (?) Pfad in $\mathcal{S}G(M)$ mit folgender Unterteilung:

$$(s, €) \overset{u}{\twoheadrightarrow} \begin{pmatrix} q, & A \\ & U \end{pmatrix} \overset{\begin{matrix} v \\ \twoheadrightarrow \\ A \\ U \end{matrix}}{} \begin{pmatrix} & A \\ q, & X \\ & A \\ & U \end{pmatrix} \overset{\begin{matrix} w \\ \twoheadrightarrow \\ A \\ X \\ A \\ U \end{matrix}}{} \begin{pmatrix} & A \\ q', & X \\ & A \\ & U \end{pmatrix} \overset{\begin{matrix} x \\ \twoheadrightarrow \\ A \\ U \end{matrix}}{} \begin{pmatrix} q', & A \\ & U \end{pmatrix} \overset{y}{\twoheadrightarrow} (f, \epsilon)$$

Tafelbild wird noch nachgereicht, dies ist erst ein Teil!

Muss zeigen: Wenn z lang genug, dann gibt es so eine Konstellation!

12.6 Definition

Ein Kapitel eines akzeptierten (?) Pfades in $\mathcal{S}G(M)$ ist eine Kellersuffixerhalnder Teilpfad maximaler Länge

$$\begin{array}{c} C \\ BBB \\ \boxed{\begin{array}{ll} A & A \\ B & \\ € & \end{array}} \end{array} \quad C$$

Beachte

1. Verschiedene Kapitel eines Pfades sind entweder verschachtelt, oder disjunkt.
2. (q, U) Knoten in akzeptierten (?) Pfad P
   $\neq$ **Kapitel**, die (q, U) enthalten, entspricht $|U|$

12.6.1 Typ eines Kapitels

$$\begin{pmatrix} q, & A \\ & U \end{pmatrix} \longrightarrow \begin{pmatrix} q', & A \\ & U \end{pmatrix} \qquad (q, A, q')$$

Die Anzahl der Typen ist $\leq |Q| \cdot |P| \cdot |Q| = t$.

$$PICTURE$$

Was passiert wenn bei Akzepterung eines langen Wertes z Stack immer $\leq t$ bleibt?

$$\Rightarrow \# \text{ der verschiedenen Status}$$

$$|Q| \sum_{0 \leq i \leq t} |\Gamma|^i = N$$

Wenn Stack ¿ t, gibt es Typenwiederholungen auf Stack!
Betrachte Punkt, wo Stack am höchsten
und betrachte oberste Typenwiederholung (muss in obersten $t+1$ Stellen passieren).
$v = x = \epsilon$ wird verhindert, indem man kürzesten akzeptierten (?) Pfad für z betrachtet.

12.7 Satz

$L \subset \Sigma^*$ sei NKA Sprache
$|\Sigma| = 1$
$\Rightarrow L$ ist regulär.

Beweis

Hilfslemma 1

$\exists N \forall z \in L$ mit $|z| > N$: $za^{N!} \in L$

Hilfskorollar

$\exists N \forall z \in L$ mit $|z| > N$: $\forall i \in \mathbb{N}: \; za^{iN!} \in L$

$$PICTURE$$

$z \in L, \; |z| > N_i$ definiere

$$m_z = \min\left\{m \in \mathbb{N} \;\middle|\; a^m \in L, \; z = a^m \cdot a^{iN!} \text{ für irgendein } i \in \mathbb{N}\right\}$$

$$M = \{m_z \;|\; z \in L, \; |z| > N\}$$

Behauptung

$$|M| \leq N!$$

$$z, z' \in L, \; |z| > N, \; |z'| > N$$

$$|z| \bmod N! = |z'| \bmod N!$$

$$\rightarrow m_z = m_{z'}$$

$$\Longleftrightarrow |z| = |z'| + kN! \quad k \in \mathbb{Z}$$

$$L = \underbrace{\underbrace{\{w \in L \;|\; |w| \leq N\}}_{\text{endlich, also regulär}} \cup \bigcup_{m \in M} \underbrace{\{a^m a^{iN!} \;|\; u \in \mathbb{N}\}}_{\text{regulär}}}_{\text{regulär}}$$

$$\underbrace{aaa \ldots a}_{m} \underbrace{(a \ldots a)^*}_{N'}$$

Beweis von Hilfslemma 1

L NKA Sprache $\Rightarrow$ L hat Pumping Eigenschaf

$$\exists N \forall z \in L\ |z| > N: \quad z = uvwxy,\ |vx| \geq 1,\ |vwx| \leq N \quad \forall i \in \mathbb{N}:\ uv^i wx^i y \in L$$

$$z = a^{|u|}a^{|v|}a^{|w|}a^{|x|}a^{|y|}$$

Wähle $i = \frac{N!}{p} + 1 \ \in \ \mathbb{N}$

Sorry, dieser Beweis wird hier abgebrochen - Null Struktur, Null Verständnis - wer ihn hat, ich freu mich dann über Post

13 Deterministische Kellerautomaten

Ein NKA ist <u>deterministisch</u> (also ein DKA),
„wenn in jeder Situation höchstens eine Regel anwendbar ist“.

$$\begin{matrix}\Sigma \\ Q \\ \Gamma \\ s \\ F \\ \in \\ \Delta\end{matrix} \qquad \forall (q, a, A) \in Q \times \underbrace{\Sigma^{\leq 1}}_{\{a,\epsilon\}} \times \Gamma$$

gibt es höchstens ein

$$(U, q') \in \Gamma^* \times Q \quad \text{mit} (q, x, A, U, q') \in \Delta$$

und

$$(q, \epsilon, A, U, q') \in \Delta \Rightarrow \forall a \in \Sigma \forall (\bar{U}, p) \subset \Gamma^* \times Q:\ (q, a, A, \bar{U}, p) \notin \Delta$$

Wollen zeigen

$$\boxed{\begin{matrix}\text{DKA} \\ \text{Sprache}\end{matrix}} \subsetneq \boxed{\begin{matrix}\text{NKA} \\ \text{Sprache}\end{matrix}}$$

also es gibt Sprache L', sodass L' NKA Sprache aber keine DKA Sprache.

Idee

1. Zeige, dass DKA Sprachen unter Komplementbildung abgeschlossen sind. Also L DKA Sprache $\Rightarrow$ $\bar{L}$ DKA Sprache
2. Gib eine NKA Sprache L an mit $\bar{L}$ nicht NKA Sprache.
   (also auch keine DKA Sprache)

$$L_{abc} = \{a^n b^n c^n \mid n \in \mathbb{N}\}$$

ist keine NKA Sprache

$$\begin{aligned} L &= \bar{L_{abc}} \\ \bar{L} &= L_{abc} \end{aligned}$$

Beweis

$$\begin{aligned} \bar{L_{abc}} &= \{a^i b^j c^k \mid i \neq j\} && \text{NKA Sprache} \\ &= \{a^i b^j c^k \mid i \neq k\} && \text{NKA Sprache} \\ &= \{w \text{ hat nicht Form } a^i b^j c^k\} && \text{regulär, also NKA Sprache} \end{aligned}$$

Idee

M DKA für L
tausche Endzustände
funktioniert „fast“

Beispiel

$$\Sigma = \{a, k\} \qquad Q = \{s, p, q, f\} \qquad \Gamma = \{€\}$$

$$\begin{aligned} \Delta: \quad & (s, a, €, €, f) \\ & (s, b, €, €, q) \\ & (f, \epsilon, €, €, p) \end{aligned}$$

akzeptiert a
aa wird nicht akzeptiert (nicht vollständig konsumiert)
b wird nicht akzeptiert, weil $q \notin F$
Es gibt Pfad:

$$(s, €) \xrightarrow{a} (f, €) \xrightarrow{\epsilon} (p, €)$$

13.1 Satz

L DKA Sprache $\Rightarrow \bar{L}$ DKA Sprache

13.2 Korrolar

$\bar{L_{abc}} \in$ NKA Sprachen $\setminus$ DKA Sprachen

13.3 Lemma 1

Für jede DKA Sprache L existiert ein DKA $\hat{M}$ mit $L(\hat{M}) = L$ und $\hat{M}$ konsumert jede Eingabe vollständig

13.4 Lemma 2

Es sei $\hat{M}$ ein DKA, der jede Eingabe vollständig akzeptiert. Aus $\hat{M}$ lässt sich ein DKA $\bar{M}$ bauen, mit $L(\bar{M}) = L(\hat{M})$.

13.4.1 Beweis: (Satz)

$$L \text{ DKA Sprache } \overset{\text{Lemma 1}}{\Longrightarrow} \exists \hat{M} \text{ mit } L(\hat{M}) = L \text{ und } \hat{M} \text{ konsumiert jede Eingabe}$$
$$\overset{\text{Lemma 2}}{\Longrightarrow} \exists \bar{M} \text{ mit } L(\bar{M}) = L(\bar{\bar{M}}) = \bar{L} \Longrightarrow \bar{L} \text{ ist DKA Sprache}$$

13.4.2 Beweis Lemma 2

$$\hat{M} = (\Sigma, \Gamma, Q, s, F€, \Delta)$$

DKA konsumfreudig!

Idee

$$\bar{M} = (\Sigma, \Gamma, Q, s, F€, \Delta)$$

Gegenbeispiel

$$\begin{array}{c} \Sigma = \{a, b\} \\ \Gamma = \{€\} \\ Q = \{s, f\} \\ F = \{f\} \end{array} \quad \Delta \quad \begin{array}{c} (s, a, €, €, f) \\ (s, b, €, €, s) \\ (f, \epsilon, €, €, s) \end{array}$$

akzeptiert alle Strings, die in a enden.

Beispiel

aba von M akzeptiert
aba wird auch von $\bar{M}$ akzeptiert

$$(s, €) \xrightarrow{a} (f, €) \xrightarrow{\epsilon(?)} (s, €) \xrightarrow{b} (s, €) \xrightarrow{a} (f, €) \xrightarrow{\epsilon} (s, €) \in Q \backslash F$$

also Endzustand von $\bar{M}$

Problem

Am Berechungsende kann es Folge von ϵ Übergängen geben, die dich Endzustände und Nicht-Endzustände führt.

Idee

Merke dir, ob seit Konsum des letzten Eingabezeichens ein Endzustand erreicht wurde

$$\bar{Q} = Q \times \{1, 2, 3\}$$

$(q, 1)$	bedeutet $\hat{M}$	ist im Zustand q seit letztem Eingabekonsum wurde ein Endzustand erreicht.
$(q, 2)$	bedeutet $\hat{M}$	ist im Zustand q seit letztem Eingabekonsum wurde KEIN Endzustand erreicht.
$(q, 3)$	bedeutet $\hat{M}$	ist im Zustand q seit letztem Eingabekonsum wurde KEIN Endzustand erreicht. und um nächsten Schritt muss ein Eingabezeichen konsumiert werden.

also es kann auch kein Endzustand über weitere ϵ Übergänge erreicht werden.

$$\bar{s} = \begin{cases} (s,1) & s \in F \\ (s,2) & s \notin F \end{cases} \qquad \bar{M} = \left(\Sigma, \Gamma, \bar{Q}, \bar{s}, \bar{F}, \in, \bar{\Delta}\right)$$

$$\bar{F} = \{(q, 3) \mid q \in Q\}$$

$$\bar{\Delta} = (q, \epsilon, A, U, p) \in \Delta$$

$p \in F$	$((q,1), \epsilon, A, U, (p,1)) \in \bar{\Delta}$
	$((q,2), \epsilon, A, U, (p,1)) \in \bar{\Delta}$
$p \notin F$	$((q,1), \epsilon, A, U, (p,1)) \in \bar{\Delta}$
	$((q,2), \epsilon, A, U, (p,1)) \in \bar{\Delta}$
()	$((q,2), \epsilon, A, A, (q,3)) \in \bar{\Delta}$
$p \in F$	$((q,3), a, A, U, (p,1)) \in \bar{\Delta}$
	$((q,1), a, A, U, (p,1)) \in \bar{\Delta}$
	$((q,2), a, A, U \ldots$ Konflikt mit ()
$p \notin F$	$((q,3), a, A, U, (p,2)) \in \bar{\Delta}$
	$((q,1), a, A, U, (p,2)) \in \bar{\Delta}$

$\bar{M}$ akzeptiert $z \in \Sigma^*$, genau dann wenn $\hat{M}$ z nicht akzeptiert.

13.4.3 Beweis zu Lemma 1

Es sei $M = (\Sigma, \Gamma, Q, s, F, \in, \Delta)$ ein DKA für L. Wie kann es sein, dass M nicht die gesamte Eingabe konsumiert?

1. es fehlt anwendbare Regel
   „Friedhofzustand“ übergehen, in dem Rest des Wortes gelesen wird
2. Keller leer
   zusätzliches neues Kellerbodensymbol, zu Friedhof, wenn je erreicht!

3. Es gibt Zyklus von ϵ Übergängen
   wenn so ein Zyklus beginnen würde, gehe zu Friedhof

$$\begin{aligned}\hat{M} =& \left(\Sigma, \hat{P}, \hat{Q}, \hat{s}, \hat{F}, \hat{\in}, \hat{\Delta}\right)\\ \hat{\Gamma} =& \Gamma \cup \left\{\hat{\in}\right\}\\ \hat{Q} =& Q \cup \left\{\hat{s}, \hat{\in}, \hat{f}\right\}\\ \hat{F} =& F \cup \left\{\hat{f}\right\}\\ \hat{\in} \notin& \Gamma\end{aligned}$$

$$\hat{\Delta} : \left(\hat{s}, \in, \in\hat{\in}, s\right) \in \hat{\Delta} \text{ zusätlicher Kellerboden}$$

$$\forall q \in Q : \left(q, \epsilon, \hat{\in}, \hat{\in}, c\right) \in \hat{\Delta} \text{ neuer Keller leer zu Friedhof}$$

$\forall q \in Q,\ \forall A \in \Gamma$:Falls in Status (q, AU)keine Regel vorhanden

$$(\forall U \in \Gamma^*)$$

$$(q, a, A, A, c) \in \hat{\Delta}$$

$$\forall a \in \Sigma\, (c, a, A, A, c) \in \hat{\Delta}$$

$$\forall A \in \Gamma$$

$\forall q \in Q,\ \forall A \in \Gamma$:Falls es aus Status (q, A)beliebig lange folgenden von ϵ Übergängen gibt

mathematisch wohldefiniert, aber nicht offensichtlich „entscheidbar“

dann

$(q, \epsilon, A, A, c) \in \hat{\Delta}$falls in der Folge kein Endzustand vorkommt.

$(q, \epsilon, A, A, \hat{f}) \in \hat{\Delta}$falls in der Folge ein Endzustand erreicht wird.

für Existenz $\hat{M}$ ist „entscheidbar“ irrelevant.

14 Abschlußeigenschaften von NKA Sprachen

14.1 Satz

$L,\ L'$ NKA Sprachen, R reguläre Sprache

1. $L \cup L'$ NKA
2. $L \cap R$ NKA
3. $L \cdot L'$ NKA
4. L^R NKA

5. $L \cap L'$ i.A. nicht NKA
6. $\bar{L}$ i.A. nicht NKA

14.2 Beweis

M	NKA für L
M'	NKA für L'
N	DEA für R

1. Baue Maschine mit neuem Startzustand $\bar{s}$ aus $\bar{s}$ ϵ Übergängen zu $\begin{matrix} s \\ \bar{s} \end{matrix}$ und M, M'
2. Idee, lasse M und N „parallel“ laufen
3. lasse M bei Akzeptanz Keller leer machen
   ϵ Übergänge von Endzuständen von M zu s'
4. „trivial“
5. $$L_{abc} = \{a^n b^n c^n \mid n \in \mathbb{N}\} \qquad \text{ist keine NKA Sprache}$$
   $$\underbrace{\{a^m b^n c^n \mid m, n \in \mathbb{N}\}}_{\text{NKA}} \qquad \underbrace{\{a^n b^n c^k \mid n, k \in \mathbb{N}\}}_{\text{NKA}}$$
6. schon bewiesen

15 Grammatiken
anderer Mechanismus zur Sprachspezifikation

$\mathcal{G}$	Σ	Sprachalphabet	„Terminale“
	V	Variablen	„Nicht Terminale“
	$S \in V$	Startsymbol	

$$P \subset (\Sigma \cup V)^* \vee (\Sigma \cup V)^* \times (\Sigma \cup V)^*$$

Produktionen, Regeln.

15.0.1 Schreibweise

$$(\alpha, \beta) \in P \qquad \text{schreibt man} \qquad \alpha \to \beta$$

15.1 Beispiel

$$\mathcal{G} = (\Sigma, V, S, P) \quad \Sigma = \{a, b, c\} \quad V = \{A, B, C, S\}$$

$$\begin{aligned} P = \{\ & S \to aSBC && 1 \\ & S \to aBC && 2 \\ & CB \to BC && 3 \\ & aB \to ab && 4 \\ & bB \to bb && 5 \\ & bC \to bc && 6 \\ & cC \to cc && 7 \\ \} & \end{aligned}$$

$\mathcal{G}$ definiert Relation auf $(V \cup \Sigma)^*$

$$u \overset{G}{\Longrightarrow} v \qquad \text{falls } u = xyz,\ v = xy'z,\ y \to y' \in P$$

„aus u leutet sich v ab“
„Ableitungsschritt“ - „Derivationsschritt“

$\overset{G}{\Longrightarrow}^*$ reflexisv transitive Hülle von $\overset{G}{\Longrightarrow}$

$$u \overset{G}{\Longrightarrow}^* v$$

heißt $\exists u_0, u_1, \ldots, u_k$ so dass $u_{i-1} \overset{G}{\Longrightarrow} u_i$ für $1 \leq i \leq k$

$$u_0 \overset{G}{\Longrightarrow} u_1 \overset{G}{\Longrightarrow} u_2 \overset{G}{\Longrightarrow} \ldots \overset{G}{\Longrightarrow} u_k$$

„Ableitung“ - „Derivation“

nach Grammatik spezifizierte Sprache

$$L(\mathcal{G}) = \left\{ w \in \Sigma^* \ \mid\ S \overset{G}{\Longrightarrow}^* w \right\}$$

15.2 Beispiel

$$
\begin{aligned}
S &\overset{G}{\Longrightarrow}_1 a\underline{S}BC \\
&\overset{G}{\Longrightarrow}_1 aa\underline{S}BCBC \\
&\overset{G}{\Longrightarrow}_2 aaa\underline{BC}\underline{BC}BC \\
&\overset{G}{\Longrightarrow}_3 aa\underline{aB}BCCBC \\
&\overset{G}{\Longrightarrow}_4 aa\underline{ab}BC\underline{CB}C \\
&\overset{G}{\Longrightarrow}_3 aaabB\underline{CB}CC \\
&\overset{G}{\Longrightarrow}_3 aaa\underline{bB}BCCC \\
&\overset{G}{\Longrightarrow}_5 aaab\underline{bB}CCC \\
&\overset{G}{\Longrightarrow}_5 aaabbbCCC \\
&\overset{G}{\Longrightarrow}_6 aaabbb\underline{cC}C \\
&\overset{G}{\Longrightarrow}_7 aaabbbcc\underline{C} \\
&\overset{G}{\Longrightarrow}_7 aaabbbccc
\end{aligned}
$$

Behauptung

$$L(\mathcal{G}) = \{a^n b^n c^n \mid n \geq 1\} = L_{abc}$$

Beweis

i)

$z \in L_{abc} \Rightarrow z \in L(\mathcal{G})$
$z = a^n b^n c^n$

Gib Ableitung für z an

$$
\begin{aligned}
S &\overset{G}{\Longrightarrow}_1 aSBC \\
&\overset{G}{\Longrightarrow}_{n-2\ Regel\ 1} \underbrace{a\ldots a}_{n-1} S \underbrace{BCBC\ldots BC}_{n-1} \\
&\overset{G}{\Longrightarrow}_{Regel\ 2} \underbrace{a\ldots a}_{n} \underbrace{BCBC\ldots BC}_{n} \\
&\overset{G}{\Longrightarrow}_{Regel\ 3} \underbrace{a\ldots a}_{n} \underbrace{B\ldots B}_{n} \underbrace{C\ldots C}_{n} \\
&\overset{G}{\Longrightarrow}_{Regel\ 4} \underbrace{a\ldots a}_{n} b \underbrace{B\ldots B}_{n-1} \underbrace{C\ldots C}_{n} \\
&\overset{G}{\Longrightarrow}_{Regel\ ?} \underbrace{a\ldots a}_{n} \underbrace{b\ldots b}_{n} \underbrace{C\ldots C}_{n} \\
&\overset{G}{\Longrightarrow}_{Regel\ 6} \underbrace{a\ldots a}_{n} \underbrace{b\ldots b}_{n} c \underbrace{C\ldots C}_{n-1} \\
&\overset{G}{\Longrightarrow}_{n-mal\ Regel\ 7} \underbrace{a\ldots a}_{n} \underbrace{b\ldots b}_{n} \underbrace{c\ldots c}_{n}
\end{aligned}
$$

i)

$Z \in L(\mathcal{G}) \Rightarrow z \in L_{abc}$, also z hat Form $a^n b^n c^n$

Behauptung 1

$S \overset{G}{\Longrightarrow}^* \alpha$, dann

$$\#_a(\alpha) = \#_b(\alpha) + \#_B(\alpha) = \#_c(\alpha) + \#_C(\alpha)$$

Beweis
Gilt für $\alpha = S$ und wird durch jeden Ableitungsschritt erhalten! (prüfe jede Regel)

Behauptung 2

$\Rightarrow z \in \Sigma^*,\ S \overset{G}{\Longrightarrow}^* z$, dann

$$\#_a(z) = \#_b(z) = \#_c(z)$$

$S \overset{G}{\Longrightarrow}^* \alpha' B \alpha'' \overset{G}{\Longrightarrow} \alpha' b \alpha''$ dann gilt $\alpha' B \alpha''$ hat Form $\underbrace{a^n b^n}_{\alpha'} B \alpha''$.

Behauptung 3

Analog für c's

15.3 Kontextfreie Grammatiken (KFG/CFG)

linke Seite jeder Produktion besteht aus genau einer Variablen

$$P \subset V \times (\Sigma \cup V)^*$$

15.3.1 Beispiel

kontextfreie Grammatik für arithmetische Ausdrücke

$$\mathcal{G} = (\Sigma, V, E, P) \qquad \Sigma = \{a, (,), *, +\}$$

$$\begin{aligned} E &\to T \\ T &\to E + T \\ T &\to F \mid T * F \\ F &\to a \mid (E) \end{aligned}$$

$$\begin{aligned} E &\Rightarrow \underline{E} + T \\ &\Rightarrow \underline{T} + T \\ &\Rightarrow \underline{T} * F + T \\ &\Rightarrow \underline{F} * F + T \\ &\Rightarrow (\underline{E}) * F + T \\ &\Rightarrow (\underline{E} + T) * F + T \\ &\Rightarrow (\underline{T} + T) * F + T \\ &\Rightarrow (\underline{F} + T) * F + T \\ &\Rightarrow (a + \underline{T}) * F + T \\ &\Rightarrow \dots \\ &\Rightarrow (a + a) * a + a \end{aligned}$$

Links Ableitung

es wird immer Produktion am linkesten nicht Terminal der schon abgeleiteten Satzform angewendet.

15.4 Syntaxbaum (Ableitungsbaum)

Baum
Knoten markiert mit Elementen von $V \cup \Sigma$

$$v \quad \text{markiert} \quad A$$

$v_1, \dots, v_k$ Kinder von v $\quad$ v_i markiert mit $\alpha_i \in (V \cup \Sigma)$, dann muss es Produktion

$$A \to \alpha_1 \alpha_2 \dots \alpha_k$$

Der String α der sich aus Markierungen der Blätter des Baumes ergibt (wenn von links nach rechts abgelesen), hat die Eigenschaft, dass $A \Rightarrow^* \alpha$, wobei A Markierung der Wurzel

15.4.1 Satz

L Sprache
$\exists$ KFG $\mathcal{G}$ mit $L(\mathcal{G}) = L \Leftrightarrow \exists$ NKA M mit $L(M) = M$

15.5 Satz

L ist NKA-Sprache $\Leftrightarrow$ L ist kontextfreie Sprache
$\exists$ NKA M mit $L(M) = L \Leftrightarrow \exists$ kontextfreie Grammatik G mit $L(G) = L$

Anmerkung

NKA Sprachen heißen daher üblicherweise kontextfreie Sprachen.

Beweis: „$\Leftarrow$“

Kontextfreie Grammatik

$$G = (\Sigma, V, S, P)$$

wollen NKA M konstruieren, sodass

$$L(M) = L(G)$$

Idee

M soll Linksableitungen „simulieren“

Terminalpräfix $\approx$ konsumierten Eingabe
Rest $\approx$ Kellerinhalt

Vorlesung Mittwoch

Also, diese Vorlesung wird noch nachgetragen ...

16 Turing Maschine

$$PICTURE$$

16.1 Formalisierung

$$
\begin{aligned}
Q &\text{ Zustandsmenge} \\
\Sigma &\text{ Eingabealphabet} \\
\Gamma &\text{ Arbeitsalphabet} \\
\bar{b} \in \Gamma &\text{ Leerzeichen} \\
s \in Q &\text{ Startzustand} \\
F \subset Q &\text{ Endzustand}
\end{aligned}
$$

$$\Delta \subset (Q \times (\Sigma \cup \{\bar{b}\}) \times \Gamma) \times (Q \times K \times \Gamma \times K)$$

16.2 Konfiguration einer Turing Maschine

- Zustand
- Eingabeband Inhalt
- Position des Eingabe Lesekopfs
- Arbeitsbandinhalt
- Position des Arbeits Lese/Schreibkopf

$$Q \times \Sigma^* \times \mathbb{Z} \times \mathcal{B} \times \mathbb{Z}$$

$$\mathcal{B} = \left\{ B : \mathbb{Z} \to \Gamma \mid \text{für nur endlich viele } i \in \mathbb{Z} \text{ gilt } B[i] \neq \bar{b} \right\}$$

16.3 Ausgangskonfiguration

$$
\begin{aligned}
init\ (x) &= (s, x, 1, B_0, 0) \\
B_0[i] &= \bar{b} \quad \forall i \in \mathbb{Z}
\end{aligned}
$$

16.4 Endkonfiguration

$$(q, \ldots,) \quad q \in F$$

16.5 Rechenschrittrelation zwischen Konfigurationen

$$(q, x, i, B, j) \overset{M}{\longmapsto} (q', x, i', B', j')$$

g.d.w.

$$\exists (q, a, A, q', \mu, A', \lambda) \in \Delta$$

mit

$$x[i] = a \qquad i' = i + \mu$$

$$B[j] = A \quad B'[l] = \begin{cases} A' & l = j \\ B[l] & l \neq j \end{cases}$$

$$j' = j + \lambda$$

$x[l] = \bar{b}$ für $l \leq 0$ und für

$\overset{M}{\longmapsto}^*$ reflexiver, transitiver Abschluss von $\overset{M}{\longmapsto}$
$x \in \Sigma^*$ wird von M akzeptiert, g.d.w.

$$init\ (x) \overset{M}{\longmapsto}^* k \qquad k \ldots \text{Endkonfiguration}$$

$$L(M) = \{x \in \Sigma^* \mid M \text{ akzetiert } x\}$$

16.6 Beispiel für Turing Maschine

$$L(M)\, \{a^n b^n c^n \mid n \in \mathbb{N}\}$$

Idee

1. Lese a's von Eingabe und kopier auf Arbeitsband
2. Lese b's von Eingabe und vergleiche mit $\#a$'s auf Arbeitsband (ziehe Arbeitskopf links!)
3. Lese c's von Eingabe und vergleiche mit $\#a$'s auf Arbeitsband (ziehe Arbeitskopf rechts!)

$$\begin{aligned}
\big(s, a, \bar{b} \quad , \quad s, +1, a, +1\big) \\
\big(s, b, \bar{b} \quad , \quad q, 0, \bar{b}, -1\big) \\
(q, b, a \quad , \quad q, +1, a, -1) \\
\big(q, c, \bar{b} \quad , \quad q', 0, \bar{b}, +1\big) \\
(q', c, a \quad , \quad q', +1, a, +1) \\
\big(q', \bar{b}, \bar{b} \quad , \quad h, 0, a, 0\big) \\
\big(s, \bar{b}, \bar{b} \quad , \quad h, 0, \bar{b}, 0\big) \\
\{h\} = F
\end{aligned}$$

17 Mehrband Turing Maschinen

PICTURE

17.1 Übergangsregeln

$$\Delta \subset (Q \times (\Sigma \cup \{\bar{b}\}) \times \Gamma^k) \times (Q \times K \times (\Gamma \times K)^k)$$

17.2 Konfigurationsmenge

$$Q \times (\Sigma^* \times \mathbb{Z}) \times (\mathcal{B} \times \mathbb{Z})^k$$

k-Band Turing Maschine können nicht mehr als 1-Band Turing Maschinen

17.3 Satz

Sei M eine Turing Maschine mit k Arbeitsbändern.
Dann gibt es eine 1-Band Turing Maschine $\hat{M}$ mit $L(\hat{M}) = L(M)$.

Beweis

Idee

Betrachte die k Bänder von M als k Spuren eines Bandes
Merke, wo die einzelnen LEseköpfe sich befinden

$$\hat{\Gamma} = (\gamma \times \{0,1\})^k$$

Für jede Regel $(q, a, A_1, \ldots, a_k, q', \mu, A_1', \lambda_1, \ldots, A_k', \lambda_k)$
Baue ein „Unterprogramm“, das den Effekt dieser Regel simuliert.
Lese/Schreibkopf von $\hat{M}$ steht am linken nde des benutzten Teils des Arbeitsbandes.

	A	B	C	A	A	B	B	C	A	
	0	0	0	0	0	0	0	0	1	
										$\bar{b}$

Für jedess von 1 bis k:
Arbeitskopf von $\hat{M}$ fährt nach rechts, bis Ez Kopfposition von M auf auf Bandstreifen i findet.
Mache Operation der Regel auf Band i nach.
Fahremit Kopf zurück zu linkem Ende.
Beobachte Initialisierung der k-Streifen Zellen am linken und rechten Ende des bearbeiteten Teils.

Beachte

Wenn M be Eingabe x $f(x)$ Rechenschritte macht, dann braucht $\hat{M} \leq \mathcal{O}(f(x)^2)$ Rechenschritte
denn, Simulation jedes der $f(x)$ Schritte von M braucht $k = \mathcal{O}(1)$ Läufe von $\hat{M}$ über den bearbeiteten Teil des Arbeitsbandes. Dieser Besteht aus höchstens $f(x)$ Zellen, da mit jedem Schritt von M nur höchstens eine weitere neue Arbeitszelle dazukommt.

Turing Maschine M heißt deterministisch, wenn in jeder Situation höchstens 1 Regel anwendbar ist.
Sonst nicht deterministisch

17.4 Satz

Sei M eine nicht deterministische Turing Maschine (mit 1 Arbeitsband).
Dann gibt es eine deterministische Turing Maschine M' mit $L(M') = L(M)$.

reguläre Sprachen $\subsetneq DKA \subsetneq$ kontextfreie $\subsetneq DTM = NTM \subsetneq$ alle Sprachen