LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Formale Sprachen und Komplexität
Sommersemester 2019

Zentralübung 25.07.2019: Wiederholung und Fragen

Prof. Dr. David Sabel

LFE Theoretische Informatik

Letzte Änderung der Folien: 25. Juli 2019

Inhaltsübersicht

Teil I: Formale Sprachen und Automatentheorie

- Chomsky-Grammatiken und die Chomsky-Hierarchie
- Reguläre Sprachen: DFAs, NFAs, reguläre Ausdrücke, Äquivalenz der Formalismen, Pumping-Lemma, Satz von Myhill-Nerode, Minimierung von DFAs, Abschlusseigenschaften, Entscheidbarkeitsresultate
- Kontextfreie Sprachen: Chomsky-Normalform, (Greibach-Normalform), Pumping-Lemma, CYK-Algorithmus, Kellerautomaten (PDA und DPDA), Abschlusseigenschaften und Entscheidbarkeitsresultate
- Kontextsensitive und Typ 0-Sprachen: Kuroda-Normalform, Turingmaschinen (DTM und NTM), LBAs, Abschlusseigenschaften

Inhaltsübersicht

Teil II: Berechenbarkeitstheorie

- Berechenbarkeit
- Turingmaschinen und Turingberechenbarkeit
- LOOP-, WHILE-, GOTO-Programme und Berechenbarkeit
- Primitiv-rekursive und μ-rekursive Funktionen
- Unentscheidbarkeit: Halteproblem
- Reduktionen, PCP, Satz von Rice

Inhaltsübersicht

Teil III: Komplexitätstheorie

- $\mathcal{P}$ und $\mathcal{NP}$
- NP-Vollständigkeit
- Polynomialzeitreduktionen
- Satz von Cook
- NP-vollständige Probleme

Eingesendete Fragen

Frage

Worin liegt der Unterschied zwischen einem Nerode-Automaten, Äquivalenzklassenautomaten und Minimalautomaten?

Sei L die akzeptierte reguläre Sprache

Nerode-Automat

DFA, der konstruiert wird als: Zustände sind Äquivalenzklassen der Nerode-Relation $\sim_L \subseteq (\Sigma^*) \times (\Sigma^*)$ wobei: $u \sim_L v \iff$ für alle $w \in \Sigma^* : uw \in L \iff vw \in L$

Äquivalenzklassenautomat

DFA, der konstruiert wird aus gegebenem DFA für L:
Zustände sind Äquivalenzklassen der Äquivalenzrelation $\equiv \subseteq (Z \times Z)$,
wobei: $z_i \equiv z_j \iff$ für alle $w \in \Sigma^* : \widehat{\delta}(z_i, w) \in E \iff \widehat{\delta}(z_j, w) \in E$

Minimalautomat

DFA, der L erkennt und eine minimale Anzahl von Zuständen hat

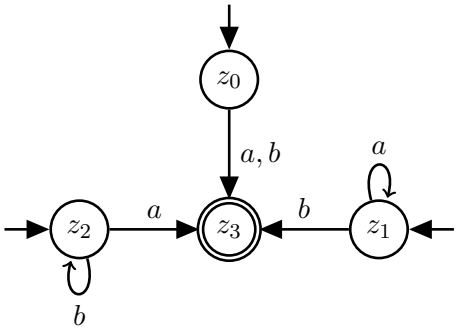
Alle drei Automaten sind isomorph (gleich bis auf Umbenennung): Bewiesen in VL

Klassiker: „Rechenaufgaben“

- Transformation NFA in DFA mit Potenzmengenkonstruktion
- Minimierung von DFAs
- Chomsky-Normalform berechnen
- CYK-Algorithmus ausführen

Beispielaufgabe zu endlichen Automaten

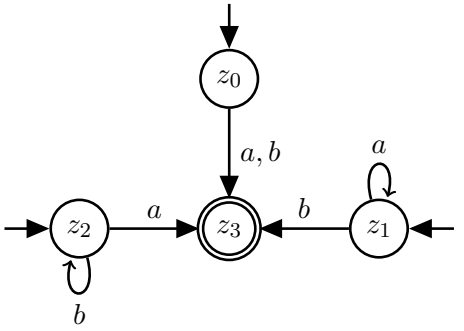
Gegeben sei der folgende NFA über $\Sigma = \{a, b\}$.



a) Welche Sprache erkennt der gezeigte NFA?
b) Geben Sie die Sprache durch einen regulären Ausdruck an.
c) Erzeugen Sie einen äquivalenten DFA durch die Potenzmengenkonstruktion (erreichbare Zustände reichen aus).
d) Minimieren Sie den DFA (Rechenweg erforderlich (Tabelle)).

Beispielaufgabe zu endlichen Automaten

Gegeben sei der folgende NFA über $\Sigma = \{a, b\}$.



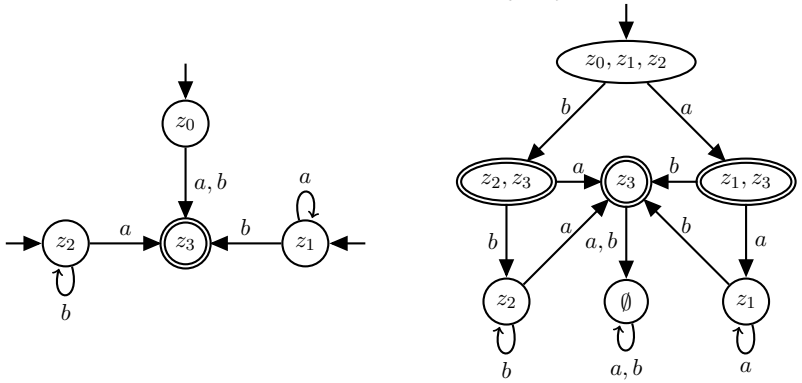
a) Welche Sprache erkennt der gezeigte NFA?

$$\begin{aligned} L &= \{a, b\} \cup \{b^i a \mid i \geq 0\} \cup \{a^i b \mid i \geq 0\} \\ &= \{b^i a \mid i \geq 0\} \cup \{a^i b \mid i \geq 0\} \end{aligned}$$

b) Geben Sie die Sprache durch einen regulären Ausdruck an.
$L = L(\alpha)$ mit $\alpha = a^*b|b^*a$

Beispielaufgabe zu endlichen Automaten

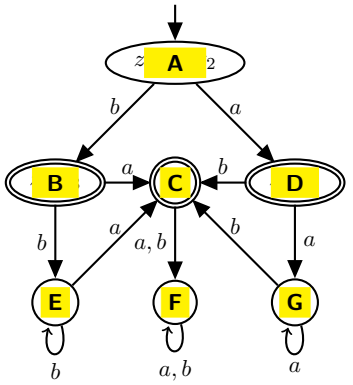
Gegeben sei der folgende NFA über $\Sigma = \{a, b\}$.



c) Erzeugen Sie einen äquivalenten DFA durch Potenzmengenkonstruktion (erreichbare Zustände reichen aus)

Beispielaufgabe zu endlichen Automaten

Gegeben sei der folgende DFA über $\Sigma = \{a, b\}$.



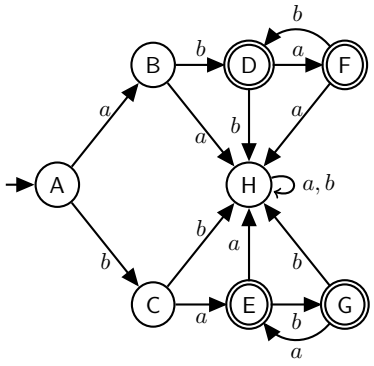
B	$\times_1$					
C	$\times_1$	$\times_2$				
D	$\times_1$	$\times_2$	$\times_2$			
E	$\times_2$	$\times_1$	$\times_1$	$\times_1$		
F	$\times_2$	$\times_1$	$\times_1$	$\times_1$	$\times_2$	
G	$\times_2$	$\times_1$	$\times_1$	$\times_1$	$\times_2$	$\times_2$
	A	B	C	D	E	F

Alle Paare nicht-äquivalent,
Automat war schon minimal!

d) Minimieren Sie den DFA (Rechenweg erforderlich (Tabelle)).

Beispielaufgabe zu Minimierung

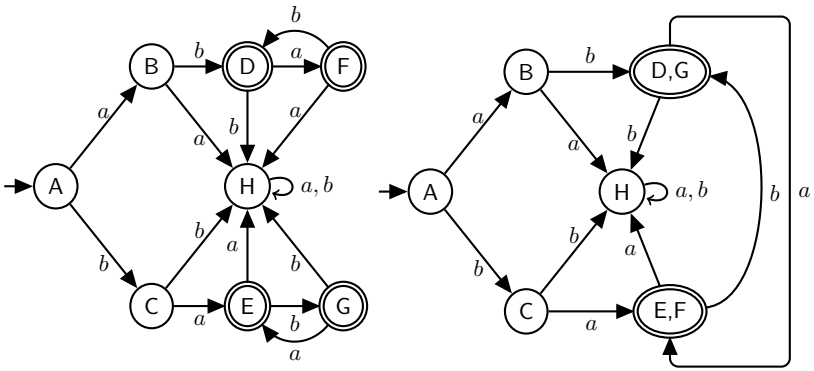
Gegeben sei der folgende DFA über $\Sigma = \{a, b\}$.



B	$\times_2$						
C	$\times_2$	$\times_2$					
D	$\times_1$	$\times_1$	$\times_1$				
E	$\times_1$	$\times_1$	$\times_1$	$\times_2$			
F	$\times_1$	$\times_1$	$\times_1$	$\times_2$			
G	$\times_1$	$\times_1$	$\times_1$		$\times_2$	$\times_2$	
H	$\times_3$	$\times_2$	$\times_2$	$\times_1$	$\times_1$	$\times_1$	$\times_1$
	A	B	C	D	E	F	G

Das Paar (D,G) und das Paar (E,F) sind äquivalent

Minimieren Sie den DFA (Rechenweg erforderlich (Tabelle)).



Chomsky-Normalform berechnen

Sei $G = (V, \Sigma, P, S)$ mit $V = \{S, A, B, C\}$, $\Sigma = \{a, b\}$
$P = \{S \to CS \mid C, A \to a, B \to b, C \to ACA \mid B\}$
Berechne die Chomsky-Normalform:

1. Entfernen von ε-Produktionen: Gibt keine.
2. Entfernen von Einheitsproduktionen: Keine Zyklen, aber Einheitsproduktionen (topologische Sortierung $S < C$)
   - Setze $B \to b$ ein: $S \to CS \mid C, A \to a, B \to b, C \to ACA \mid b$
   - Setze $C \to ACA$ und $C \to b$ ein:
     $S \to CS \mid ACA \mid b,\ A \to a, B \to b, C \to ACA \mid b$
3. Abkürzungen einf. (wir verwenden $A \to a$ und $B \to b$ wieder):
   $S \to CS \mid ACA \mid b,\ A \to a, B \to b, C \to ACA \mid b$
4. Rechte Seiten auf 2 Variablen kürzen:
   $S \to CS \mid AN_1 \mid b, A \to a, B \to b, C \to AN_2 \mid b,$
   $N_1 \to CA, N_2 \to CA$

CYK

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

- $V = \{S, A, B, C\}$
- $\Sigma = \{a, b\}$
- $P = \{S \to CS \mid b, A \to a, B \to b, C \to AC \mid b\}$

Führe den CYK-Algorithmus für $aaaaab$ aus. Liegt das Wort in $L(G)$?

	a	a	a	a	a	b
	1	2	3	4	5	6
1	A	A	A	A	A	B,C,S
2					C	
3				C		
4			C			
5		C				
6	C					

Da unten links nicht das Startsymbol S in der Tabelle steht, liegt das Wort nicht in $L(G)$

CYK

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

- $V = \{S, A, B, C\}$
- $\Sigma = \{a, b\}$
- $P = \{S \to CS \mid b, A \to a, B \to b, C \to AC \mid b\}$

Führe den CYK-Algorithmus für $aaaaabb$ aus. Liegt das Wort in $L(G)$?

	a	a	a	a	a	b	b
	1	2	3	4	5	6	7
1	A	A	A	A	A	B,C,S	B,C,S
2					C	S	
3				C	S		
4			C	S			
5		C	S				
6	C	S					
7	S						

Da unten links das Startsymbol S in der Tabelle steht, liegt das Wort $L(G)$

Klassiker: „Beweisaufgaben“

- Nichtregularität einer Sprache zeigen mit Pumping-Lemma
- Nichtregularität einer Sprache zeigen mit Satz von Myhill-Nerode
- Nicht-Kontextfreiheit einer Sprache zeigen mit Pumping-Lemma für CFLs
- Unentscheidbarkeit zeigen mit Reduktion
- Unentscheidbarkeit zeigen mit Satz von Rice
- NP-Vollständigkeit zeigen, u.a. mit Polynomialzeitreduktionen

Nichtregularität beweisen

Aufgabe

Zeige $L = \{a^j\$b^j \mid j \in \mathbb{N}\}$ ist nicht-regulär.

Mit dem Pumping-Lemma:

- Sei $n > 0$ beliebig.
- Sei $z = a^n\$b^n$. Dann gilt $|z| \geq n$ und $z \in L$.
- Sei $z = uvw$ ein beliebige Zerlegung von z mit $|uv| \leq n$ und $|v| \geq 1$.
- Damit muss gelten $u = a^i$, $v = a^j$, $w = a^k\$b^n$ mit $i + j + k = n$ und $j \geq 1$. Dann gilt $uv^0w = a^{n-j}\$b^n \notin L$.

Somit erfüllt L die Pumping-Eigenschaft nicht und kann daher auch nicht regulär sein.

Nichtregularität beweisen

Aufgabe

Zeige $L = \{a^j\$b^j \mid j \in \mathbb{N}\}$ ist nicht-regulär durch Verwendung des Satzes von Myhill und Nerode.

Es gilt $[a^j\$]_{\sim_L} \neq [a^i\$]_{\sim_L}$ für alle $i \neq j$:

- für $w = b^i$ gilt $a^i\$w \in L$, aber $a^j\$w \notin L$
- damit $a^j\$ \not\sim_L a^i\$$ für $i \neq j$.
- Es gibt unendlich viele disjunkte Äquivalenzklassen bez. $\sim_L$: $[a^1\$]_{\sim_L}$, $[a^2\$]_{\sim_L}$, $[a^3\$]_{\sim_L}$, ...
- Das zeigt: Index($\sim_L$) $= \infty$.
- Der Satz von Myhill-Nerode zeigt nun, dass L nicht regulär sein kann.

Kontextfreiheit widerlegen

Aufgabe

Zeige $L = \{ab^iab^iab^iab^i \mid i \in \mathbb{N}\}$ ist nicht kontextfrei.

- Sei $n > 0$ beliebig.
- Sei $z = \underbrace{ab^n}_{r_1}\underbrace{ab^n}_{r_2}\underbrace{ab^n}_{r_3}\underbrace{ab^n}_{r_4}$. Dann ist $|z| \geq n$ und $z \in L$.
- Sei $z = uvwxy$ eine beliebige Zerlegung mit $|vwx| \leq n$ und $|vx| > 0$
- Dann kann vwx nur Teilwort von r_1r_2, r_2r_3 oder r_3r_4 sein.
- Wenn v oder x ein a enthält, dann $uv^2wx^2y \notin L$, da es mehr als 4 a's enthält
- Wenn v und x kein a enthalten, dann ist $uv^0wx^0y \notin L$, da das Löschen von b's in 1-2 der Teilworte r_1, r_2, r_3, r_4 passiert. D.h. es gibt noch ab^n (mindestens 2) aber es gibt auch ab^j mit $j < n$
- Daher erfüllt L die Pumping-Eigenschaft für CLFs nicht.
- Das Pumping-Lemma für CFLs zeigt nun, dass L nicht kontextfrei ist.

Kontextfreiheit widerlegen

Aufgabe

Zeigen Sie, dass $H_0 = \{w \mid \text{TM } M_w \text{ hält bei leerer Eingabe}\}$ nicht kontextfrei ist.

Pumping-Lemma?

Braucht man nicht:

- H_0 ist unentscheidbar
- Daher ist H_0 nicht einmal von Typ 1
- Daher ist H_0 auch nicht von Typ 2.

Unentscheidbarkeit zeigen

Aufgabe

Sei $X = \{w \mid M_w \text{ hält genau bei Eingabe } x\ \}$.
Zeigen Sie die Unentscheidbarkeit mithilfe einer Reduktion.

Sei $H_0 = \{w \mid M_w \text{ hält bei leerer Eingabe}\}$.

Aus der VL: H_0 ist unentscheidbar.

Wir zeigen $H_0 \leq X$:

Die Reduktionsfunktion f nimmt eine Turingmaschinenbeschreibung und erstellt daraus eine neue Turingmaschinenbeschreibung.

Sei w ein Wort. Wenn w keine Turingmaschinenbeschreibung ist, dann sei $f(w) = w$. Ansonsten sei M_w die Turingmaschine zu w.

f erstellt daraus eine Turingmaschine T, sodass ...

Unentscheidbarkeit zeigen

Aufgabe

Sei $X = \{w \mid M_w \text{ hält genau bei Eingabe } x\ \}$.
Zeigen Sie die Unentscheidbarkeit mithilfe einer Reduktion.

- T prüft, ob die Eingabe x ist. Falls nicht, geht T in eine Endlosschleife.
- T löscht das Eingabeband.
- T simuliert M_w bei leerer Eingabe.
- Wenn M_w akzeptiert, dann akzeptiert T ansonsten läuft T endlos.

Es gilt: $w \in H_0$
g.d.w M_w hält bei leerer Eingabe
g.d.w. $T = M_{f(w)}$ hält bei Eingabe x
g.d.w. $f(w) \in X$

Da f total und berechenbar ist, gilt $H_0 \leq X$.

Unentscheidbarkeit zeigen

Aufgabe

Sei $X = \{w \mid M_w \text{ hält genau bei Eingabe } x\ \}$.
Zeigen Sie die Unentscheidbarkeit mit dem Satz von Rice.

Sei $S = \{f \in \mathcal{R} \mid f(x) \text{ ist definiert, } f(u) = \perp \text{ für } u \neq x\}$
Dann ist

$$\begin{aligned} C(S) &= \{w \mid M_w \text{ berechnet eine Funktion aus S }\} \\ &= \{M_w \text{ akzeptiert bei Eingabe } x \text{ und hält nicht für} \\ &\quad \text{andere Eingaben}\} \end{aligned}$$

nach dem Satz von Rice unentscheidbar.

Weitere typische Aufgaben

- Sprachen angeben in einem der vielen Formalismen: DFA, NFA, regulärer Ausdruck, reguläre Grammatik Kontextfreie Grammatik, Kellerautomat, deterministischer Kellerautomat Turingmaschine angeben
- Formalismen ineinander überführen, z.B. regulärer Ausdruck in DFA usw.
- Programme schreiben als: Turingmaschine, WHILE-, LOOP-, GOTO-Programm, primitiv rekursive Funktion, μ-rekursive Funktion
- Sprachen „typisieren“ in der Chomsky-Hierarchie
- Wissensfragen

LOOP-Programm

Aufgabe

Seien P, Q, R LOOP-Programme. Geben Sie ein LOOP-Programm an, dass folgende Funktion berechnet:

$$jenachdem(x) = \begin{cases} P, \text{wenn } x = 0 \\ Q, \text{wenn } x = 1 \\ R, \text{sonst} \end{cases}$$

$x_P := 1;$
$x_Q := 2;$
$x_R := 1;$
LOOP x **DO** $x_P := x_P - 1; x_Q := x_Q - 1$ **END**;
LOOP x_P **DO** $x_Q := 0; x_R := 0; P$ **END**;
LOOP x_Q **DO** $x_R := 0; Q$ **END**;
LOOP x_R **DO** R **END**

Primitiv rekursive Funktionen

Aufgabe

Seien p, q, r primitiv rekursive Funktion. Geben Sie ein LOOP-Programm an, dass folgende Funktion berechnet:

$$jenachdem(x) = \begin{cases} p(x), \text{wenn } x = 0 \\ q(x), \text{wenn } x = 1 \\ r(x), \text{sonst} \end{cases}$$

Sei
$$\begin{aligned} jenachdem(x) &= f_1(x) \\ f_1(x) &= \begin{cases} g_1() & \text{wenn } x = 0 \\ h_1(f_1(x-1), x-1) & \text{sonst} \end{cases} \\ g_1 &= p(0) \\ h_1(a,b) &= f_2(b) \\ f_2(x) &= \begin{cases} g_2() & \text{wenn } x = 0 \\ h_2(f_2(x-1), x-1) & \text{sonst} \end{cases} \\ g_2 &= q(1) \\ h_2(a,b) &= r(succ(succ(b))) \end{aligned}$$

Aufgabe

Geben Sie eine DTM über $\Sigma = \{1, 2\}$ an, die auf dem Eingabeband jede 1 durch eine 2 ersetzt und dann akzeptiert.

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ mit

- $Z = \{z_0, z_1\}$
- $\Sigma = \{1, 2\}$
- $\Gamma = \Sigma \cup \{\square\}$
- $\delta(z_0, 2) = (z_0, 2, R)$
  $\delta(z_0, 1) = (z_0, 2, R)$
  $\delta(z_0, \square) = (z_1, \square, N)$
- $E = \{z_1\}$

Kellerautomat

Aufgabe

Geben Sie einen deterministischen Kellerautomat an, der $\{a^i ccb^i \mid i \in \mathbb{N}\}$ erkennt.

Sei $K = (Z, \Sigma, \Gamma, \delta, z_0, \#)$ mit

- $Z = \{z_0\}$
- $\Sigma = \{a, b, c\}$
- $\Gamma = \{\#, A\}$
- $\delta(z_0, a, \#) = \{(z_0, A\#)\}$
  $\delta(z_0, a, A) = \{(z_0, AA)\}$
  $\delta(z_0, c, \#) = \{(z_1, \#)\}$
  $\delta(z_0, c, A) = \{(z_1, A)\}$
  $\delta(z_1, c, \#) = \{(z_2, \#)\}$
  $\delta(z_1, c, A) = \{(z_2, A)\}$
  $\delta(z_2, b, A) = \{(z_2, \varepsilon)\}$
  $\delta(z_2, \varepsilon, \#) = \{(z_2, \varepsilon, \varepsilon)\}$
  und $\delta(z_i, x, X) = \emptyset$ für alle anderen Fälle.