

afe

$\Sigma^*$  endliches Alphabet,  $\Sigma^*$  alle Wörter über Alphabet,  $\epsilon$  leeres Wort,  $\cdot$  Konkatenation  
 $L \subseteq \Sigma^*$  Sprache

reguläre Sprache Abgeschlossen unter  $\cap, \cup, -, \epsilon^*, \cdot$

reguläre Ausdrücke (Denotation)

Terminologien

$\rightarrow \emptyset$

$\rightarrow a \in \Sigma$

$\alpha, \beta \rightarrow (\alpha \cdot \beta) \in \text{Reg}(\Sigma)$

$\alpha, \beta \in \text{Reg}(\Sigma) \rightarrow (\alpha \cdot \beta) \in \text{Reg}(\Sigma)$

$\alpha \in \text{Reg}(\Sigma) \rightarrow \alpha^* \in \text{Reg}(\Sigma)$

Semantik

$L(\emptyset) = \emptyset$

$L(a) = \{\epsilon a\}$  für jedes  $a \in \Sigma$

$L(\alpha + \beta) = L(\alpha) \cup L(\beta)$

$L(\alpha \cdot \beta) = L(\alpha) \cdot L(\beta)$

$L(\alpha^*) = (L(\alpha))^*$

Endliche Automaten (prozedural)

Transitionssysteme zum Checken von Sprachzugehörigkeit

$S = (\Sigma, Q, \Delta)$

Alphabet / Übergangsrelation  
endliche  
nichtleere  
Zustandsmenge

Anfangs- / akzeptierende  
Zustand / Zustände  $A \subseteq Q$   
nicht dafür  
nicht  
nicht  
nicht  
Übergangsfunktion  
 $Q \times \Sigma \rightarrow Q$

Deterministische endliche Automaten (DFA)

$A = (\Sigma, Q, q_0, \delta, A)$

Anfangs- / akzeptierende  
Zustand / Zustände  $A \subseteq Q$   
nicht dafür  
nicht  
nicht  
Übergangsfunktion  
 $Q \times \Sigma \rightarrow Q$

Für jedes  $q$  und  $w$  muss eine Transition  $\delta(q, w)$  existieren

akzeptierte Sprache  $L(A) = \{w \in \Sigma^* : A \text{ akzeptiert } w\}$

$\delta : Q \times \Sigma^* \rightarrow Q$

Produktautomat

$Q := Q^1 \times Q^2$

$q_0 := (q_0^1, q_0^2)$

$S((q_1, q_2), w) := (S^{(1)}(q_1, w), S^{(2)}(q_2, w))$

$V : A := (A^1 \times A^2) \cup (Q^1 \times A^2)$

$\Lambda : A := A^1 \times A^2$

Komplement  
 $A := Q \setminus A_{\text{alt}}$

Nichtdeterministische endliche Automaten (NFA)

$A = (\Sigma, Q, q_0, \Delta, A)$

$\Delta \subseteq Q \times \Sigma \times Q$  keine mehrere Transitionen auch möglich  
keine Epsilontransitions

NAEFA

$L(A) = L(A^{\text{det}})$

NFA  $\rightarrow$  DFA Potenzmengentrück

Sei  $A = (\Sigma, Q, q_0, \Delta, A)$

$A^{\text{det}} = (\Sigma, \hat{Q}, \hat{q}_0, \hat{\delta}, \hat{A})$

$\hat{Q} = \{S \subseteq Q : S \cap A \neq \emptyset\}$   
 $\hat{\delta}(S, a) = \{q' \in Q : (q, a, q') \in \Delta \text{ für min ein } q \in S\}$

Beweis Induktion (EW Anfang  $i=0$ ;  $w(i) = a_1 \dots a_i$ )

$S_i = \text{Erreichbare Zustände}$   
 $S_{i+1} \xrightarrow{a_i} S_i$

NFA

Konkatenation

$Q := Q^1 \cup Q^2$  (disjunkt)

$q_0 := q_0^{(1)}$

$A := A^{(2)}$

$\Delta := \Delta^{(1)} \cup \Delta^{(2)} \cup \{(\epsilon, q_1, q_2) : q \in Q^1 \text{ und } q_1, q_2 \in A\}$

Sturm Operation analog  $\rightarrow$  für ein  $q \in A^{(1)}$

||

regulärer Ausdruck  $\Rightarrow$  NFA

Beweis Induktion

Kleene'sches Theorem

Satz von Kleene

reg. Ausdr. äquivalent zu DFA/NFA

## Satz von Kleene

reg. Ausdruck, DFA, NFA äquivalent

DFA  $\rightarrow$  regulärer Ausdruck

Sei  $A = (\Sigma, Q, q_0, \delta, F)$ ,  $Q = \{q_1, \dots, q_m\}$

Induktion

$L_{i,m}^k$  Sprache der Wörter mit Lauf  $l$ ,  $q_1, \dots, q_s$  in  $m$  auf  $A$  nur mit Zuständen  $\leq k$

$$\text{Anfang } L_{i,m}^0 = \begin{cases} \{\epsilon \in \Sigma : \delta(q_0, \epsilon) = q_i\} & \text{falls } l \neq m \\ \{\epsilon \in \Sigma : \delta(q_0, \epsilon) = q_i\} \cup \{\epsilon \in \Sigma : \delta(q_l, \epsilon) = q_i \text{ und } l = m\} & \text{(inkl } \epsilon\text{)} \end{cases} \quad \text{(Buchstaben die transition von } t \text{ zu } q_i \text{ haben)}$$

$$L_{i,m}^{k+1} = L_{i,m}^k \cup L_{i,k+1}^k \cdot (L_{k+2, k+1}^k)^* \cdot L_{k+1, m}^k$$

Operationen unter denen reg. Ausdruck abgeschlossen, daher auch  $\sigma_{i,m}^k$

Minimiert Äquivalenzrelation  $\sim_L$

$\sim_L$  Äquivalenzrelation  $w \sim_L w' \Leftrightarrow \forall x \in \Sigma^*: wx \in L \Leftrightarrow w'x \in L$

rechtsinvariant:  $w \sim_L w' \Rightarrow \forall u \in \Sigma^*, wu \sim_L w'u$

auf Automat  $w \sim_A w' \Leftrightarrow \hat{\delta}(q_0, w) = \hat{\delta}(q_0, w')$  Verfeinerung von  $\sim_L \Rightarrow \text{index}(\sim_L) \leq \text{index}(\sim_A)$   
 $\text{index}(\sim_A) = |\Omega|$  endlich.

Satz von Myhill-Nerode

$\sim_L$  hat endlichen Index  $\Rightarrow L$  regulär

$\Rightarrow$  Beweis durch Äquivalenzklassen automat.

$$A = (\Sigma^*/\sim_L, [\epsilon], \delta, \{[q] : q \in F\}) \quad \text{(Beweis Induktio...}$$

$\delta([q], a) = [qa]$

Minimautomat bis auf Isomorphismus eindeutig (DFA)

$\Rightarrow$  Äquivalenzklassenautomat  $L$  Form  $A = (Q, (\delta_q)_{q \in Q}, A, q_0)$

$$\delta_q(a) = \delta(q, a)$$

Isomorphismus bijektive Abbildung  $f: Q^{(1)} \rightarrow Q^{(2)}$   
mit  $f(q^{(1)}) = q^{(2)}$

$$\begin{aligned} f(q \in Q^{(1)}) &= f(\delta^{(1)}(q, a)) = \delta^{(2)}(f(q), a) \\ f[A^{(1)}] &= A^{(2)} \end{aligned}$$

Minimierung:

1. alle unerreichbaren Zustände entfernen

2. sukzessive Verfeinerung  $\sim$

$q \sim q' \Leftrightarrow (q \in A \wedge q' \in A) \vee (q \notin A \wedge q' \notin A)$ , daher Klassen  $A, Q \setminus A$

$q \sim q' \Leftrightarrow q \sim q' \vee \exists a \in \Sigma : \delta(q, a) \neq \delta(q', a)$

BRUNNEN nach endlichen Schritten:  $\sim = \sim_n = \sim_{n+1}$

$$\Rightarrow q \sim q' \Leftrightarrow \hat{\delta}(q_0, q) \cap A = \hat{\delta}(q_0, q') \cap A$$

~~$q \sim q' \Leftrightarrow \hat{\delta}(q_0, q) = \hat{\delta}(q_0, q')$~~

$$q = \hat{\delta}(q_0, q)$$

$$q' = \hat{\delta}(q_0, q')$$

## Pumping Lemma für reguläre Sprachen

(Bereis Zustand in DFA muss sich wiederholen)

Für jede reguläre Sprache  $L \subseteq \Sigma^*$   $\exists n \in \mathbb{N} \quad \forall x \in L : |x| \geq n \quad \exists u =uvw : v \neq \epsilon \quad \forall m \in \mathbb{N} \quad uv^m w \in L$   
z.B.  $a^nb^n$  pumpbar aber nicht regulär

Lektion 3

## Algorithmische Fragen

Menge zulässiger Eingaben

Entscheidungsproblem für  $x \in I$  hat Eigenschaft D $\hat{2}$   
daher  $x \in D$  Dals DSI  
oder  $x \in I \setminus D$

Zugehörigkeit zu regulärer Sprache: DFA linearer Entscheidungsalgorithmus

Leerheitsproblem: min 1 akzeptierender Zustand in DFA erreichbar

Sprachgleichheit/Aquivalenzproblem: Rückführung  $L_1 \setminus L_2$  und  $L_2 \setminus L_1$  leer?

Ausdruckskomplexität: reg. Ausdruck  $\rightarrow$  NFA: Größe NFA poly nomal zu Länge Ausdruck beschränkt (kann z.B. durch Zulassung Komplement  $?$   $\rightarrow$  DFA in der Regel exponentiell wachsen)  
 $DFA \rightarrow$  reg. Ausdruck: exponentiell

## Grammatiken (Formalisierung Erzeugungsprozess)

Konvention Großbuchstaben  
 ↗ Kleinbuchstaben  
 $G = (\Sigma, V, P, X_0)$

endliches Terminalalphabet  $\Sigma \neq \emptyset$   
 Produktionsregeln  $P \subseteq (V \cup \Sigma)^+ \times (V \cup \Sigma)^*$

endliche Variablenmenge  $V \cap \Sigma = \emptyset$

Für  $p = (v, v') \in P$  auch  $v \rightarrow v'$   
 erlaubt Ersetzung von  $v$  zu  $v'$  linke Seite rechte Seite

Ableitbarkeit:  
 $x \xrightarrow{G} x' \Leftrightarrow x = uvw, x' = u'v'w$  und Produktion  $v \rightarrow v'$   
 $x \xrightarrow{* G} x' \Leftrightarrow$  endliche Ableitungssequenz (ein  $\xrightarrow{G}$ -Schritt)  
 reflexiv, transitiv

Angabe  $P$  oft als  $P = \{(v_i, v'_i) : i \in \mathbb{N}\}$

$v_1 \rightarrow v_1'$  Backus-Naur Form (BNF) erwartet auch

$v_n \rightarrow v_n'$   $v \rightarrow v_1' | v_2' | \dots | v_n'$   $x \rightarrow u[v]w$  für  $x \rightarrow uw$   
 stattdessen auch ::= möglich/üblich  $x \rightarrow u[v]w$  für  $x \rightarrow uw$   $z \in \Sigma | v$

$L(G) = \{w \in \Sigma^*: X_0 \xrightarrow{* G} w\}$  |  $L(G^2) = L(G^2) \Leftrightarrow G^2$  äquivalent zu  $G^2$

## Chomsky Hierarchy

relevante Komplexität Wortproblem jeweiliger Sprache als Entscheidungsproblem (z.B.  $|v'| \geq |v|?$ )  
 harmlose  $\epsilon$ -Produktion, wenn  $X_0 \rightarrow \epsilon$  und  $X_0$  nur als Startsymbol vorkommt

Typ 3  
regulär alle Produktionen rechtslinear, d.h. von Form  $x \rightarrow e, x \rightarrow a$  oder  $x \rightarrow ay$  reg. Ausdrücke DFA, NFA  $V \cap \Sigma^*$

Typ 2  
kontextfrei nur Produktionen der Form  $X \rightarrow r$

Typ 1  
kontextsensitiv nur harmlose  $\epsilon$ -Produktionen  
alle anderen Verkürzung  $v \rightarrow v'$  mit  $|v'| \geq |v|$

Typ 0  
allgemein keine Einschränkungen

Lemma: zu jeder kontextfreien Grammatik gibt es eine äquivalente kontextsensitive

Übertragung Chomsky auf Sprachen für diese strikt

Kontextfreie Sprachen abgeschlossen unter:  $U, \cdot, *$  nicht unter  $\cap, \lambda, -$

### Chomsky-Normalform

nur Produktionen der Form:  $X \rightarrow YZ, X \rightarrow a$

Jede kontextfreie Grammatik ohne  $\epsilon$ -Transitionen ist äquivalent zu einer in Chomsky Normalform

Beweis: 1. alle Terminalymbole durch neue Variable ersetzen 2.  $X \rightarrow Y$  eliminieren 3.  $X \rightarrow Y, Y \rightarrow Z$  eliminieren durch Zusatzvariable

Länge Ableitung immer:  $2|w|-1$

Ableitungsbaukasten binär, d.h. min:  $|w| \geq 2^d$  d-Tiefe

### Pumping Lemma.

Für jede kontextfreie Sprache  $L \subseteq \Sigma^*$   $\exists n \in \mathbb{N} \forall x \in L: |x| \geq n \exists x = yuvwz: u, v, w \in \Sigma^*$

$u, v, w, z \in \Sigma^*$

$yv^mrv^mz \in L$

### CYK Algorithmus

Teilwort  $u_{i,j} = a_i \dots a_j$

$X \xrightarrow{*} u_{i,j} \Leftrightarrow i=j \wedge (X \rightarrow a_i) \text{ FP}$   
 $v \neq j \wedge X \rightarrow YZ \text{ FP mit } Y \xrightarrow{*} u_{i,k} \wedge Z \xrightarrow{*} u_{k+1,j} \exists i \leq k \leq j$

# Berechnungsmodelle zur Präzision des abstrakten Berechnungsbegriffe

PDA Push Down Automat (erweitert NFA) äquivalent zu kontextfrei

$$P = (\Sigma, Q, q_0, \Delta, \Gamma, \#, \epsilon)$$

Anfangskettensymbol  
 Kettensymbol  
 $\Delta \subseteq Q \times \Gamma \times (\Sigma \cup \{\epsilon\}) \times \Gamma^* \times Q$   
 startes Kettensymbol  
 eingesetzene Produktions oder  
 neue geprägte Kettensymbole

Restkett  
Ketten  
Konfiguration  $C = (q, v, \alpha) \in Q \times \Sigma^* \times \Gamma^*$   
Start:  $C_0 = (q_0, w, \#)$   $v = w$   
für  $(q, r, \beta, q') \in \Delta$   $C' = (q', r, \beta)$   
 $\alpha' = \beta \alpha_{rest}$   
Akzeptiert:  $C = (q, \epsilon, \epsilon)$   $q \in A$

$$L(P) = \{w \in \Sigma^*: C_0[w] \xrightarrow{P} (q, \epsilon, \epsilon) \text{ für ein } q \in A\}$$

Grammatik  $\Rightarrow$  PDA

$$\text{Sei } G = (\Sigma, V, P, X_0) \quad \Sigma \cap V = \emptyset$$

$$P = (\Sigma, \{\epsilon\}, Q, \Delta, \{\epsilon\}, V \cup \Sigma, X_0)$$

$(q, X, \epsilon, \alpha, q)$  für jede Produktion  $X \Rightarrow \alpha$  von  $G$

$(q, a, a, \epsilon, q)$  für jedes  $a \in \Sigma$

PDA  $\Rightarrow$  Grammatik

$$\text{Sei } P = (\Sigma, Q, q_0, \Delta, \Gamma, \#, \epsilon)$$

$$G = (\Sigma, V, P, X_0)$$

$P:$

$$V := (Q \times \Gamma \times Q) \cup \{X_0\}$$

$$X_0 \xrightarrow{P} (q_0, H, q) \quad \text{für } q \in A$$

$$(q, r, q) \xrightarrow{P} x \quad \text{für } (q, r, x, \epsilon, q')$$

$$(q, r, q) \xrightarrow{P} x \quad (q_1, r_1, q_1) \quad \text{für } (q, r, x, r_1, q_1) \in \Delta$$

$$(q, r, q) \xrightarrow{P} x \quad (q_1, r_1, q_1) (q_2, r_2, q_2) \dots (q_n, r_n, q_n) \quad \text{für } (q, r, x, r_1, q_1) \in \Delta$$

Turingmaschine universelles Berechnungsmodell (informelle Vorstellung was prinzipiell berechenbar ist)

Church-Turing These: Turingmaschinen können alle erdenklichen Algorithmen erfassen

funktionalle Anforderungen an Algorithmus:

Sequentialität und Definitheit (Schrittweise abarbeiten eindeutiger Einzelanweisungen)  
endliche Kontrolle/Steuerung  
Lokalität (lokaler Zugriff auf Daten in Einzelschritten)

deterministische Turingmaschine (DTM) für Wortproblem

$$M = (\Sigma, Q, q_0, \delta, q^+, q^-)$$

ende  
zustands-  
menge  
verwerfender Endzustand  $q^+ \neq q^-$   
akzeptierender Zustand

$$\text{Startzustand: } Q \times (\Sigma \cup \{\square\}) \rightarrow (\Sigma \cup \{\square\}) \times \{\leftarrow, \downarrow, \uparrow, \rightarrow, \square\}$$

- $w \xrightarrow{M} \alpha$  Berechnung auf  $M$  divergiert  
 $w \xrightarrow{M} \text{STOP}$  terminiert  
 $w \xrightarrow{M} q^+$  akzeptiert Eingabe  
 $w \xrightarrow{M} q^-$  verwirft Eingabe

entscheidbar/  
aufzählbar  
entscheidbar  
löst Wortproblem  
Entscheidungsproblem

Sprache  
 $M$  akzeptiert  $L(M) = \{w \in \Sigma^*: M$  akzeptiert  $w\}$   
 $M$  entscheidet Sprache  $L$ , wenn  $M$  auf  $\Sigma^*$  terminiert  
und  $L = L(M)$

$\langle M \rangle \cdot \Sigma^*$  Kodierung von  $M$

Haltproblem unentscheidbar

zulässige Eingaben  $I = \{\langle M \rangle : M \text{ DTM über } \Sigma\}$   
 $H = \{\langle M \rangle : \langle M \rangle \xrightarrow{M} \text{STOP}\} \subseteq I$

Nachweis Unentscheidbarkeit durch Reduktion auf Haltproblem

z.B. Äquivalenzproblem zweier Programme (ob PDA nicht entscheidbar)

Kettierungsproblem: quadratische fortigerketten beliebig groß pfasterbar?

Arithmetik: z.B. Nullstellen über Polynom in  $\mathbb{Z}$

nicht unter Kompilatoren abgeschlossen

Chomsky von Typ 0  $\Leftrightarrow$  wird von DTM akzeptiert

Typ 1  $\Rightarrow$  von DTM entscheidbar (nutzt nicht verkürzend und Zyklen eliminierbar)

unter allen abgeschlossen

Es gibt auch nicht aufzählbare Sprachen

Alle Inklusionen der Chomsky-Hierarchie für Sprachen sind strikt

Typ 1  $\Leftrightarrow$  nichtdeterministische, linear platzbeschränkte Turing Maschine

unter allen Operationen abgeschlossen

Kopf  
↓  
 $(= (\alpha, q, x, \beta) f(\Sigma \cup \{\square\})^* \times \delta \times (\Sigma \cup \{\square\}) \times \{r\})$   
links von Kopf  
↑  
rechts von Kopf

$$\langle D, w \rangle = (\Sigma, q_0, \square, w)$$

Sei  $\delta(q, x) = (x', d, q')$

$$\begin{cases} (\alpha, q^+, x, \beta) & \text{falls } d = 0 \\ (\alpha, q^+, x', \beta) & \text{falls } d = c \\ (\alpha, x', q^+, b, \beta, 0) & \text{falls } d = r \end{cases}$$

$$\alpha q = \alpha$$

$$b \beta_0 = \beta$$

Lentscheidbar  $\Leftrightarrow \frac{L}{I}$  aufzählbar

Beweis durch Widerspruch

Annahme  $M_0$  löst Haltproblem d.h.  
falls:  $\langle M \rangle \xrightarrow{M_0} \text{stop} \Rightarrow \langle M \rangle \xrightarrow{M_0} q^+$   
falls:  $\langle M \rangle \xrightarrow{M_0} \text{oo} \Rightarrow \langle M \rangle \xrightarrow{M_0} q^-$

Konstruiere  $M_1$ :

divergiert, wenn  $M_0$  akzeptiert

verwirft, wenn  $M_0$  verwirft

Widerspruch

$$\langle M_0 \rangle \xrightarrow{M_1} \text{oo} \Leftrightarrow \langle M_1 \rangle \xrightarrow{M_0} \text{stop}$$

$$L_n(B) = \{v \in (\Sigma V)^*: |v|_B \leq n\}$$

sukzessiv bestimbar

ausgangs

oder Automaten, die vorwärts und rückwärts

und überschreiben können

## Aufgabentypen

$$(A \setminus (B \cap C)) = (A \setminus B) \cup (A \setminus C)$$

$\subseteq$  Sei  $x \notin A \setminus (B \cap C)$ . Dann ist  $x \in A$  und  $x \notin B \cap C$ . Also ist  $x \in A \setminus B$  oder  $x \in A \setminus C$  und somit  $x \in (A \setminus B) \cup (A \setminus C)$ .

Äquivalenzrelation: reflexiv, symmetrisch, transitiv

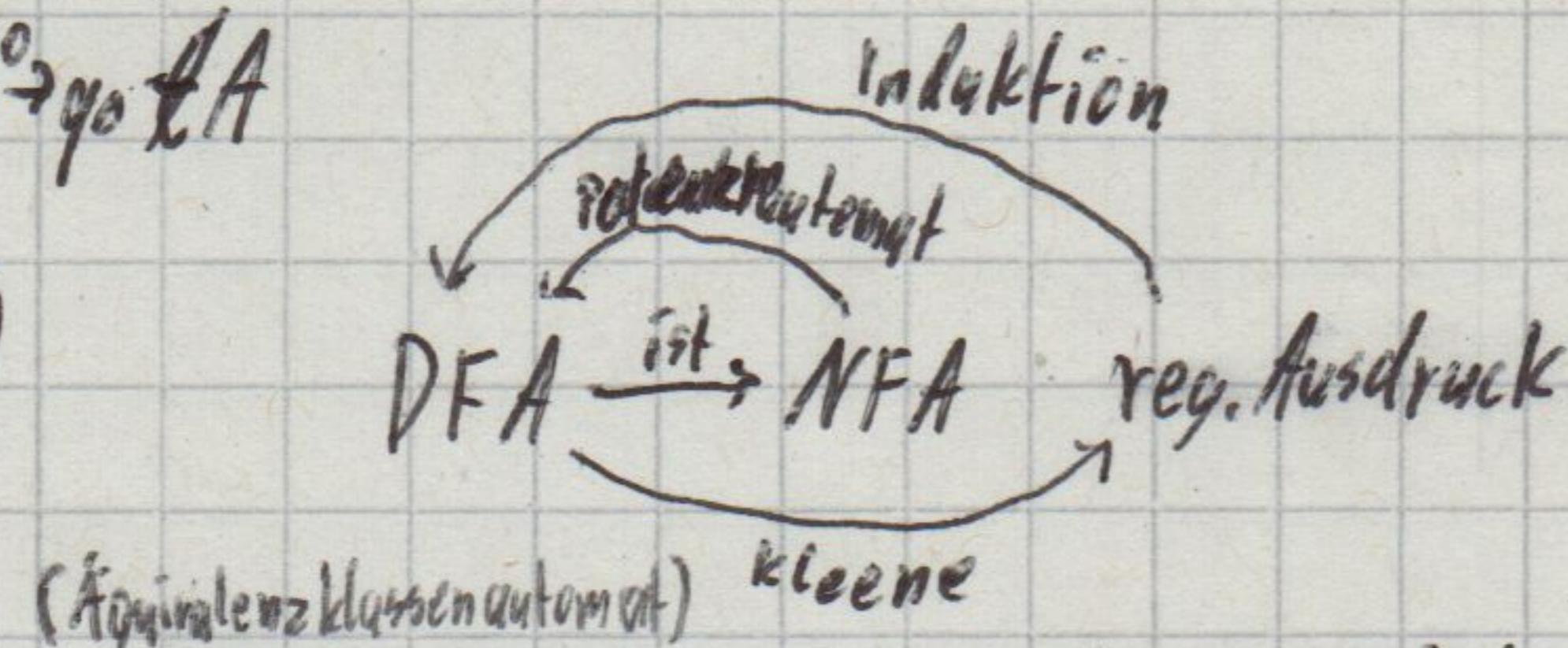
Monoid (Assoziativ, neutrales)

strukturelle Induktion: Anfang  $P(E, w)$   
Schritt  $P(a u), w$   $\stackrel{=}{\rightarrow}$  LH.

Automaten Lauf:  $q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{c} q_3 \xrightarrow{d} q_4 \notin A$

$$NFA = (\Sigma, Q, q_0, \Delta, A)$$

$$DFA = (\Sigma, Q, q_0, \delta, A)$$



DFA  $\&$  Operationen

Produktautomat  $\rightarrow$  skizzieren

Potenzmengentrick

Minimieren

	a	b	
$p_1$	$p_1(a)$	$p_1(b)$	$p_1^m$
$p_2$	$p_2(a)$	$p_2(b)$	$p_2^m$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$p_n$	$p_n(a)$	$p_n(b)$	$p_n^m$

Eckzustände

$l = l_1 \dots l_n$

Zeigen Sie  $L \subseteq M^* \Rightarrow L^* \subseteq M^*$ :  $(\epsilon L)^* \subseteq L_{l_1, \dots, l_n} \cap L$

$$L = \underbrace{(m_{1,1} \dots m_{1,k_1})}_{l_1} \dots \underbrace{(\dots \dots m_{n,1} \dots m_{n,k_n})}_{l_n} \in M^*$$

$$(L_1 L_2)^{n+1} = \underbrace{(L_1 L_2) \dots (L_1 L_2)}_{n+1-\text{mal}} = L_1 \underbrace{(L_2 L_1) \dots (L_2 L_1)}_{n-\text{mal}} L_2 = L_1 (L_1 L_2)^n L_2$$

nicht reguläre Sprachen

- Pumping Lemma: Sei  $n \in \mathbb{N}$  beliebig und betrachte das Wort  $x = \dots$  mit  $|x| \geq n$  und  $x \in L$

Sei  $x = uvw$  mit  $|v| \leq n$  und  $|v| > 0$ . Dann ist  $v$  von der Form  $v = \dots$  mit  $0 \leq k \leq n$

Für  $m=0$  enthält  $uv^m w = uw$  deshalb ...

Das widerspricht  $uw \notin L$ . Wir folgern, dass  $L$  das Pumping Lemma verletzt und nicht regulär sein kann.

- Myhill-Nerode: Wir zeigen, dass der Index von  $\sim_L$  nicht endlich und  $L$  nach Myhill-Nerode nicht regulär ist.

Betrachte die Folge von Wörtern  $(w_n)_{n \in \mathbb{N}}$  mit  $w_n = \dots$ . Sei  $n \in \mathbb{N}$  beliebig und  $m \in \mathbb{N}$  mit  $m < n$ .

Es gilt  $w_n \neq_L w_m$ , da  $w_n \in L$  aber  $w_m \notin L$ . Folglich sind alle Wörter der Folge  $(w_n)_{n \in \mathbb{N}}$  paarweise nicht  $\sim_L$ -äquivalent, was impliziert, dass der Index von  $\sim_L$  nicht endlich ist.

$$G = (\Sigma, \Delta, P, p_0, X_0) \quad P: X_0 \rightarrow \dots$$

$$L(G) \subseteq L$$

Wir nutzen Induktion über die Länge der Ableitung von  $w$ , um zu zeigen, dass  $w \in L$ , wenn  $X \xrightarrow{G}^* w$ . Dies gilt offensichtlich für  $w = \epsilon$ , da  $X \xrightarrow{G} \epsilon$  und  $\epsilon \in L$ .

Im Induktionsgeschritt sei  $|w| \geq 1$  und  $X \xrightarrow{G}^* w$ ; wir unterscheiden  $\dots$  Fälle:

-  $w = avbw$  mit  $X \xrightarrow{G}^* v$  und  $X \xrightarrow{G}^* w$ ; also gilt nach Induktionsvoraussetzung  $v, w \in L$   
z.B. über  $1 \cdot 1a$  z.B. jedes Präfix von

$L(G) \supseteq L$  Algorithmus zur Ableitung von  $w \in L$

BRUNNEN

z.B. für Wort  $w = ababc$  lautet: Beschriften  $b$ , beschriften  $a$ , konsolidierender  $a$   
Induktion hat Form  $w = a^n a; a b; b \dots$  Nach 1. Voraussetzung  $X \xrightarrow{G}^* a^n a$   
und damit  $X \xrightarrow{G} a^n a b c$  also insgesamt  $X \xrightarrow{G}^* a^n a b c$

In Chomsky einteilen: Nachweis  $L$  in Niveau, Nachweis nicht zu einfacherem Niveau

Chomsky-Normalform

1. Schritt: eliminieren  $\epsilon$ -Produktionen
2. Schritt: neue Variablen für Buchstaben
3. Schritt: eliminieren  $X \rightarrow Y$  und  $X \rightarrow X_0 \dots X_k$  mit  $k \geq 2$

CYK-Algorithmus

Länge	b	a	u	b	a
1	B		A, C		
2					
3					
4					
5					

$S \xrightarrow{G} ABIBC$   
 $B \xrightarrow{G} BA$

$L(B) = \{a^n b^n c^n : n \in \mathbb{N}\}$

$b = (\epsilon, \epsilon x, xz, pxz)$

$P: x \rightarrow \epsilon$

$L = \{a^p : p \text{ Prim}\}$

nicht kontextfrei, da Pumping Lemma

Sei  $n \in \mathbb{N}$ . Wir betrachten  $x = a^p : p \geq n+2 \text{ prim}$ . Nach dem Lemma gibt es eine

Zerlegung  $x = yuvwz$  mit  $|uvw| \geq 1$ ,  $|uvw| \leq n$ , ~~und~~  $yv^m w^m z \notin L$  für jedes  $m \in \mathbb{N}$ .

Sei  $k := |uvw|$ ,  $m := |vz| > 1$ . Dann hat  $yv^m w^m z$  die Länge  $m + mk = (k+1)m$ , was nicht prim ist. Widerspruch!