Formale Sprachen II

Vorlesung: Michael Ludwig, Sebastian Schöner
Mitschrieb: Jan-Peter Hohloch

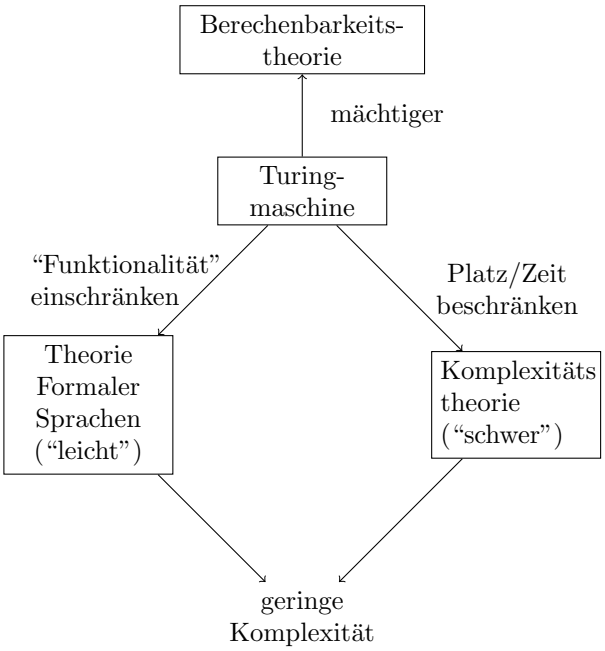
WS 15/16

Inhaltsverzeichnis

1 Einführung

1.1 Formale Sprachen

- Was ist die Theorie Formaler Sprachen?
  - schlecht definierter Begriff!
- Sei Σ eine *endliche* Menge, die wir Alphabet nennen
- Σ^* ist die Menge der *endlichen* Sequenzen von Σ-Elementen
- Eine (formale) Sprache ist die Teilmenge von Σ^*
- Es gibt $2^{|\Sigma^*|} > |\mathbb{N}|$ Sprachen, uns interessieren nur abzählbar viele davon.
- Wir stellen Sprachen durch endliche Beschreibungen dar
  - Grammatik
  - Automaten (Endlich, Keller, Turingmaschine,...)
  - Logik
  - Algebra
  - Schaltkreise
  - Ausdrücke
  - ...



1.2 Wiederholung Theoretische Informatik

1.2.1 Chomsky-Hierarchie

Alle Sprachen
$\supset$ Typ 0 (Turingmaschine)
$\supset$ Typ 1 (kontextsensitiv) ($\mathcal{O}(n)$ Platz)
$\supset$ Typ 2 (kontextfrei) (Kellerautomat)
$\supset$ Typ 3 (regulär) (endlicher Automat)

1.2.2 Berechenbarkeit

- Church-Turing-These
  - Sprache entscheidbar $\Leftrightarrow$ charakteristische Funktion berechenbar ($\Rightarrow$ Wortproblem, andere lassen sich darauf abbilden)
- Reduktion: $A \leq_m B :\Leftrightarrow (\exists f : x \in A \Leftrightarrow f(x) \in B)$ (many-one Reduktion)
- Reduktion (Turing): $A \leq_T B :\Leftrightarrow \exists TM$ mit Orakel B, die A erkennt.

1.2.3 Komplexitätstheorie

- L ist C-schwer $:\Leftrightarrow \forall X \in C : X \leq L$
- L ist C-vollständig $:\Leftrightarrow$ L ist C-schwer und L$\in$ C

1.2.4 Theorie formaler Sprachen

Reguläre Sprachen

- REG = Typ3-Grammatik = DEA = NEA = reguläre Ausdrücke = endliche Monoide
- NEA $\rightarrow$ DEA: Potenzmengenkostruktion
- DEA: Minimalautomat, vgl. Myhill-Nerode
- Abschlüsse: Komplement, Schnitt, Vereinigung, Konkatenation, Stern, Homomorphismen, inverse Homomorphismen, Quotienten, ...
- Entscheidbarkeit: Wort, Leerheit, Äquivalenz, Endlichkeit
- Tradeoff: Abschluss und Entscheidbarkeit vs. Ausdrucksstärke
- Pumping Lemma

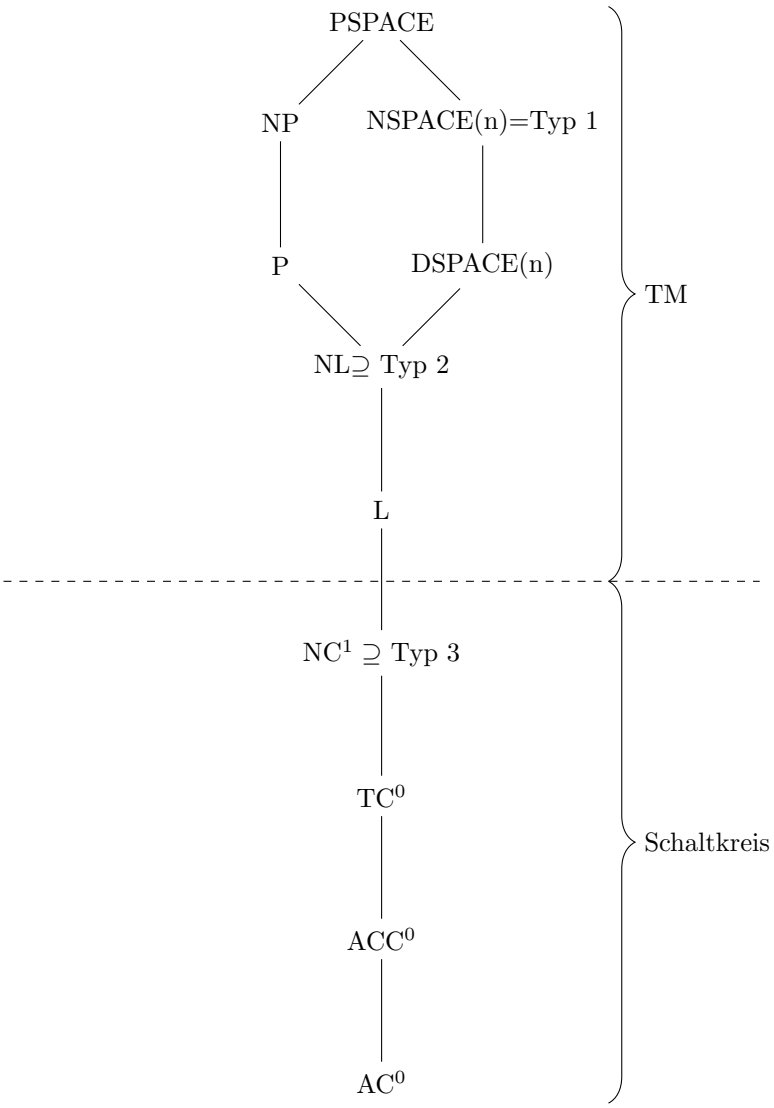


Abbildung 1.1: Komplexitätsklassen

Kontextfreie Sprachen

- Kellerautomaten = Typ 2-Grammatiken
- Wortproblem: CYK $\mathcal{O}(n^3)$ Laufzeit
- CNF: $A \to BC,\ A \to a$ ($\Rightarrow$ binärer Ableitungsbaum)
- Abschlüsse: Vereinigung, Konkatenation, Stern
- nicht abgeschlossen unter: Komplement, Schnitt
- Entscheidbarkeit: Wortproblem, Leerheit
- nicht entscheidbar: Äquivalenzproblem
- Pumping Lemma

1.3 Wiederholung Formale Sprachen I

1.3.1 Algebra

- Homomorphismus: $\varphi(w_1 w_2) = \varphi(w_1)\varphi(w_2)$
- Monoid: abgeschlossen, assoziativ und neutrales Element existiert
- $\varphi : \Sigma^* \to M,\ X \subseteq M$, dann ist $\varphi^{-1}(X)$ eine Sprache
  oder: $L \subseteq \Sigma^*$ wird von M erkannt $\Leftrightarrow L = \varphi^{-1}(\varphi(L))$
- L regulär $\Leftrightarrow$ L wird von endlichem Monoid erkannt.

1.3.2 Logik

- Beispiel:
  - $\exists x \forall y : y \leq x \wedge Q_a(x) \in FO[\leq]$ (“Wort endet auf a”)
  - $\exists X \ldots$ (monadische Logik 2. Ordnung (MSO))
- später mehr

2 Visibly Pushdown Languages (VPL)

Alur, Madhusudan 2004 / Mehlhorn 1980 (input driven PDA)

2.1 Motivation

- Tradeoff: Ausdrucksstärke vs. Abschlusseigenschaften, Entscheidbarkeit
- guter Tradeoff: VPL
- PDA (Kellerautomat): Akzeptiert per leerem Keller
- DPDA (deterministischer Kellerautomat): Akzeptiert per Endzustand
- VPA ist eingeschränkter DPDA:
  - Eingabezeichen bestimmen Kelleraktion
  - Partitionierung des Alphabets:

    $\Sigma = \Sigma_{push} \dot\cup \Sigma_{pop} \dot\cup \Sigma_{neutral}$

    $\hat\Sigma = (\Sigma_{push}, \Sigma_{pop}, \Sigma_{neutral})$

    push = call, pop = return, neutral = internal
- Beispiel: $\hat\Sigma = (\{a\}, \{b\}, \{c\})$
  - $a^n b^n$
  - $(a^n b^n)^*$
  - $\{w \mid |w|_a = |w|_b\} \notin VPL$

    $\rightarrow$ bei negativem Keller nie in VPL
- Denkmodell: $\Delta w = |w|_{\Sigma_{push}} - |w|_{\Sigma_{pop}}$

  $\rightarrow$ hat "Zacken", muss am Ende des Wortes wieder bei 0 sein

2.2 Definition: VPA

$M = (Q, Q_S, Q_F, \Gamma, \hat\Sigma, \delta, \#)$ mit:

- Q: Zustände
- Q_S: Startzustände

- Q_F: Endzustände
- Γ: Kelleralphabet
- $\hat{\Sigma}$: Eingabealphabet
- $\#$: Kellerbodenzeichen
- δ: Übergänge

$$\delta \subseteq Q \times \Sigma_{neutral} \times Q$$
$$\cup\, Q \times \Sigma_{push} \times \Gamma \setminus \{\#\} \times Q$$
$$\cup\, Q \times \Sigma_{pop} \times \Gamma \times Q$$

2.3 Semantik

- Konfiguration: Element $Q \times \hat{\Sigma}^* \times \Gamma^*$
- Konfigurationsübergänge: Relation $\rightarrow \subseteq K \times K$
- $c \in \Sigma_{neutral}:\ q, w_1, \ldots, w_n, \gamma_1, \ldots, \gamma_k \rightarrow \delta(q,c), w_1, \ldots, w_{n-1}, \gamma_1, \ldots, \gamma_k$
- $a \in \Sigma_{push}:\ q, w_1, \ldots, w_n, \gamma_1, \ldots, \gamma_k \rightarrow q', w_1, \ldots, w_{n-1}, \gamma_1, \ldots, \gamma_k, \gamma$
  $(q', \gamma) \in \delta(q, a)$
- $b \in \Sigma_{pop}:\ q, w_1, \ldots, w_n, \gamma_1, \ldots, \gamma_k \rightarrow q', w_1, \ldots, w_{n-1}, \gamma_1, \ldots, \gamma_{k-1}$
  $q' \in \delta(q, b, \gamma_k)$
- $L(M) := \{w \mid \exists s \in Q_S \exists f \in Q_F : s, w, \# \rightarrow^* f, \varepsilon, \#$ oder $\in \Gamma^*\#\}$

2.4 Alur-Kongruenz

2.4.1 Wiederholung: Myhill-Nerode

Myhill-Nerode-Relation in Abhängigkeit von $L \subseteq \Sigma^*$
$u \sim_L v :\Leftrightarrow \forall x, y : xuy \in L \Leftrightarrow xvy \in L$ (syntaktische Kongruenz)
$\Sigma^*/\sim_L = \{[w] | w \in \Sigma^*\}$ heißt syntaktisches Monoid

Anmerkungen

- $[w_1][w_2] = [w_1 w_2]$
- L regulär $\Leftrightarrow |\Sigma^*/\sim_L|$ endlich

2.4.2 Definition

$u, v \in WM$ (WM: Well-matched $\rightarrow$ akzeptieren mit leerem Keller im Endzustand)
$u \simeq v :\Leftrightarrow \forall x, y \in \Sigma^* : xuy \in L \Leftrightarrow xvy \in L$

2.4.3 Satz

L ist VPL $\Leftrightarrow |WM/\simeq|$ endlich

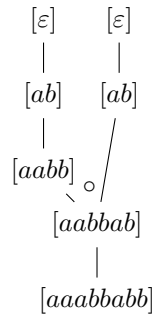
2.4.4 Anmerkungen

- Operationen
  - Konkatenation
  - Extend:

    $$[w] \to [awb],\ a \in \Sigma_{push}, b \in \Sigma_{pop}$$
- $WM := \{w \in \Sigma^* \mid \Delta(w) = 0 \wedge \forall i \leq |w| : \Delta(w_1 \dots w_i) \geq 0\}$
  mit $\Delta(w) = |w|_{\Sigma_{push}} - |w|_{\Sigma_{pop}}$
- $\Sigma^* \supseteq L$ regulär $\to \hat{\Sigma} = (\emptyset, \emptyset, \Sigma)$
- Jedes visibly Wort lässt sich durch concat und extend erzeugen

Beispiel

Leite $aaabbabb$ ab:



2.5 Satz

Für jeden VPA M existiert DVPA M' mit L(M)=L(M') wobei L(M)⊆WM

Beweis

geg. $M = (Q, Q_S, Q_F, \Gamma, \hat{\Sigma}, \delta, \#)$, konstruiere $M' = (Q', q'_S, Q'_F, \Gamma', \hat{\Sigma}, \delta', \#)$

- $Q' = \mathcal{P}(Q \times Q)$
- $q'_S = \{(q, q) \mid q \in Q\} \in Q'$ $(= id_Q)$
- $Q'_F = \{S \in Q' \mid \exists s \in Q_S \exists f \in Q_F : (s, f) \in S\}$
- $\Gamma' = \Sigma_{push} \times Q' \cup \{\#\}$

- δ':
  - $a \in \Sigma_{push} : \delta'(S, a) = (id_Q, (a, S))$
  - $b \in \Sigma_{pop} : \delta'(S, b, (a, S')) = S''$
    $S'' = \{(q, q') \mid \exists q_1, q_2, q_3 : (q, q_1) \in S', \exists \gamma \in \Gamma : (q_2, \gamma) \in \delta(q_1, a), (q_2, q_3) \in S, q' \in \delta(q_3, b, \gamma))\}$
  - $c \in \Sigma_{neutral} : \delta'(S, c) = \{(q, q') \mid \exists q'' \in Q : (q, q'') \in S \wedge q' \in \delta(q'', c)\}$

2.6 Satz von Chomsky-Schützenberger, 1963

Jede kontextfreie Sprache ist das homomorphe Bild einer Sprache $\mathbb{D}_{2k} \cap R$ wobei R regulär.
$\forall L \subseteq \Sigma^*$ kf. $\exists k \in \mathbb{N}, R$ reg. $\exists \varphi : L = \varphi(\mathbb{D}_{2k} \cap R)$ $\varphi : \Sigma^* \to \{[^1_1,]^1_1, [^1_2,]^1_2, \dots\}^*$

2.6.1 Beweis

L kf $\Rightarrow \exists G = (V, \Sigma, P, S)$ mit $L(G) = L$ und L in CNF
$x \in L(G) \Leftrightarrow \exists$ Ableitungsbaum (mit CNF: Binärbaum)
Idee: Nutze Korrespondenz zwischen Bäumen und Dyck-Sprachen.
Konstruiere Grammatik $G' = (V', \Sigma', P', S')$ wobei $L(G') = \mathbb{D}_{2k} \cap R$

Sei $P = \{\Pi_1, \dots, \Pi_k\}$, dann $P' = \{\Pi'_1, \dots, \Pi'_k\}$

- ist $\Pi_i = A \to BC$, dann $\Pi'_i = A \to \underset{i}{\overset{1}{[}} B \underset{i}{\overset{1}{]}} \underset{i}{\overset{2}{[}} C \underset{i}{\overset{2}{]}}$
- ist $\Pi_i = A \to a$, dann $\Pi'_i = A \to \underset{i}{\overset{1}{[}} \underset{i}{\overset{1}{]}} \underset{i}{\overset{2}{[}} \underset{i}{\overset{2}{]}}$

$\Sigma' = \{\underset{i}{\overset{1}{[}}, \underset{i}{\overset{1}{]}}, \underset{i}{\overset{2}{[}}, \underset{i}{\overset{2}{]}}, \dots, \underset{k}{\overset{1}{[}}, \underset{k}{\overset{1}{]}}, \underset{k}{\overset{2}{[}}, \underset{k}{\overset{2}{]}}\}$
Wähle $\varphi : \Sigma' \to \Sigma$:

- Ist $\Pi_i = A \to BC$, so $\varphi(\underset{i}{\overset{1}{[}}) = \varphi(\underset{i}{\overset{1}{]}}) = \varphi(\underset{i}{\overset{2}{[}}) = \varphi(\underset{i}{\overset{2}{]}}) = \varepsilon$
- Ist $\Pi_i = A \to a$, so $\varphi(\underset{i}{\overset{1}{]}}) = \varphi(\underset{i}{\overset{2}{[}}) = \varphi(\underset{i}{\overset{2}{]}}) = \varepsilon, \varphi(\underset{i}{\overset{1}{[}}) = a$

1) $L = \varphi(L(G'))$

2) $L(G') = \mathbb{D}_{2k} \cap R$

1) $x \in L \Leftrightarrow x \in L(G)$
   $\Leftrightarrow \exists$ Ableitungsbaum von G für x
   $\Leftrightarrow x' \in L(G')$, wobei x' Klammerrepräsentation des Baumes ist
   $\Leftrightarrow x \in \varphi(L(G'))$, da für jedes Blatt das richtige Zeichen übrig bleibt

2) offensichtlich: $L(G') \subseteq \mathbb{D}_{2k}$
Weitere Bedingungen an die Wörter, um Konstruktion zu genügen:

a) einem $\overset{1}{\underset{i}{]}}$ muss immer $\overset{2}{\underset{i}{[}}$ folgen

b) einem $\overset{2}{\underset{i}{]}}$ muss $\overset{1}{\underset{j}{]}}$ oder $\overset{2}{\underset{j}{]}}$ folgen (beachte Sonderfall für Startvariable)

c) ist $\Pi_i = A \to BC$, dann muss $\overset{1}{\underset{i}{[}}$ ein $\overset{1}{\underset{p}{[}}$ folgen mit $\Pi_p = B \to EF$ oder $\Pi_p = B \to b$
und $\overset{2}{\underset{i}{[}}$ muss immer $\overset{1}{\underset{q}{[}}$ folgen mit $\Pi_q = C \to GH$ oder $\Pi_q = C \to c$

d) ist $\Pi_i = A \to a$, dann muss auf $\overset{1}{\underset{i}{[}}\,\overset{1}{\underset{i}{]}}$ folgen und auf $\overset{2}{\underset{i}{[}}\,\overset{2}{\underset{i}{]}}$ und auf $\overset{1}{\underset{i}{]}}\,\overset{2}{\underset{i}{[}}$

e) $\overset{1}{\underset{i}{[}}$ als erstes Zeichen, dann muss Π_i S auf der linken Seite haben

→ Bedingungen sind alle regulär. Schneide alle Sprachen dieser Bedingungen; erhalte R, regulär.

2.7 Satz von Greibach

"Die schwerste kontextfreie Sprache"

2.7.1 Satz (Greibach, 1974)

Es gibt eine feste Sprache L_0, sodass L kontextfrei $\Leftrightarrow \exists$ Homomorphismus $\varphi : L = \varphi^{-1}(L_0)$

Anmerkungen

- Homomorphismen sind leicht zu berechnende Reduktionen
- Interpretation: Kann eine Maschine L_0 entscheiden, so auch jede andere kontextfreie Sprache
- L kontextfrei $\Rightarrow \exists G : L = L(G),\ G$ in Greibachnormalform (GNF):
  $A \to aA_1A_2 \dots A_k$

Beweis

Idee: Wähle φ und L_0 so, dass sich Ableitung in GNF in L_0 wiederfindet.

$\Sigma = \{a_1, a_2, b_1, b_2, c, d, \$\},\ L_0 \subseteq \Sigma^*,\ x_i, z_i \in \Sigma^*, y_1y_2 \dots y_n \in \$\mathbb{D}_2$

$L_0 = \{\epsilon\} \cup \{x_1 c y_1 c z_1 d \dots x_0 c y_n c z_n d | n \geq 1; x_i, z_i \in \Sigma^*; y_1, y_2, \dots, y_n \in \$\mathbb{D}_2\}$
L_0 ist kontextfrei.

Sei G in GNF mit $L = L(G)$ und $G = \left(\{A_1, \dots, A_{|V|}\}, \Gamma, P = \{\Pi_1, \dots, \Pi_k\}, A_1\right)$, $P_a = \{\Pi \in P \mid \Pi \text{ beginnt rechts mit } a\}$, $\forall a \in \Gamma$

$\varphi : \Gamma^* \to \Sigma^*, a \mapsto \left(\prod_{\Pi \in P_a} c\mathcal{T}(\Pi)\right) cd = c\mathcal{T}(\Pi_{P_1}) c\mathcal{T}(\Pi_{P_2}) \dots c\mathcal{T}(\Pi_{|P_a|}) cd$

$\Pi_j = A_i \to a A_{j_1} \dots A_{j_m}$

$\mathcal{T}(\Pi_j) = \underbrace{b_1 b_2^i b_1}_{A_i} \underbrace{a_1 a_2^{j_m} a_1}_{A_{j_m}} \dots \underbrace{a_1 a_2^{j_1} a_1}_{A_{j_1}}$

Sonderfall:
$\Pi_j = A_1 \to \dots \Rightarrow \mathcal{T}(\Pi_j) = \$a_1 a_2^1 a_1 \mathcal{T}(\Pi_j)$ (wie oben)
Anm: Startvariable kommt nur links vor

Beispiel

$$\Pi_1 = A_1 \to a A_2 A_2 \qquad \mathcal{T}(\Pi_1) = \$a_1 a_2^1 a_1 b_1 b_2^1 b_1 a_1 a_2^2 a_1 a_1 a_2^2 a_1$$
$$\Pi_2 = A_2 \to a \qquad \mathcal{T}(\Pi_2) = b_1 b_2^2 b_1$$
$$\Pi_3 = A_2 \to b A_2 \qquad \mathcal{T}(\Pi_3) = b_1 b_2^2 b_1 a_1 a_2^2 a_2$$
$$\Pi_4 = A_2 \to b \qquad \mathcal{T}(\Pi_4) = b_1 b_2^2 c b_1$$

$\varphi(a) = c\mathcal{T}(\Pi_1) c\mathcal{T}(\Pi_2) cd$
$\varphi(b) = c\mathcal{T}(\Pi_3) c\mathcal{T}(\Pi_4) cd$
$A_1 \to_G^* aabA_2$

$$\begin{aligned}\varphi(aab) =& c\mathcal{T}(\Pi_1) c\mathcal{T}(\Pi_2) cdc\mathcal{T}(\Pi_1) c\mathcal{T}(\Pi_2) cdc\mathcal{T}(\Pi_3) c\mathcal{T}(\Pi_4) cd \\ =& c\$a_1 a_2^1 a_1 b_1 b_2^1 b_1 a_1 a_2^2 a_1 a_1 a_2^2 a_1 c b_1 b_2^2 b_1 cd \\ & c\$a_1 a_2^1 a_1 b_1 b_2^1 b_1 a_1 a_2^2 a_1 a_1 a_2^2 a_1 c b_1 b_2^2 b_1 cd \\ & c b_1 b_2^2 b_1 a_1 a_2^2 a_1 c b_1 b_2^2 b_1 cd\end{aligned}$$

Ableitung $aabA_2$:

$$A_1 \overset{\Pi_1}{\Rightarrow} aA_2A_2 \overset{\Pi_2}{\Rightarrow} aaA_2 \overset{\Pi_3}{\Rightarrow} aabA_2 \dots$$

$y_1 y_2 y_3$ "kürzen"$\to \$a_1 a_2^2 a_1 = \mu(y_1 y_2 y_3)$ (μ entfernt nebeneinander stehende zusammenpassende "Klammern"; y sind nur die Ableitungs-relevanten Teile, der Rest steckt in x_i und z_i)
$y_4 = \mathcal{T}(\Pi_4) \to y_1 y_2 y_3 y_4 \in \$\mathbb{D}_2$ Leite in Satzform immer die Variable am weitesten links ab.
Allgemein:
$(*) A_1' \to aB_1 \dots B_m$
$A_1 \to_G^* a_1 \dots a_k A_1' \dots A_n \overset{(*)}{\to} a_1 \dots a_k a B_1 \dots B_m A_2 \dots A_n$
$A_1 \bar{A}_1 \dots \mid A_n' \dots A_1' \rightsquigarrow A_n' \dots A_1' \bar{A}_1' B_m \dots B_1$

Lemma

$A_1 \to_G^* a_1 \dots a_k A_{i1} \dots A_{ir}$ mit Produktionen $(\Pi_{p1}, \dots, \Pi_{pk})$
$\Leftrightarrow$

(a) $\varphi(a_1 \dots a_k) = x_1 c y_1 c z_1 d \dots x_k c y_k c z_k d$
x_i, z_i: nicht benutzte Produktionen

(b) $y_1 \dots y_k \in Prefix(\$\mathbb{D}_2)$

(c) $\mu(y_1 \dots y_k) = \$ a_1 a_2^{i_r} a_1 \dots a_1 a_2^{i_1} a_2$

also $A_1 \to^* a_1 \dots a_k \Leftrightarrow (a) \wedge y_1 \dots y_n \in \$\mathbb{D}_2 \wedge \mu(y_1 \dots y_n) = \$$

Beweis

durch Induktion über k

IA: $A_1 \to_G^1 a_1 A_{i_1} \dots A_{i_r}$ dann
$\exists \Pi = A_1 \to a_1 A_{i_1} \dots A_{i_r} \in P_{a_1}$
$\varphi(a_1) = x_1 c y_1 c z_1 d = c\mathcal{T}(\Pi_{p1}) c \mathcal{T}(\Pi_{p2}) c \dots c\mathcal{T}(\Pi_{pm}) d$
$\mathcal{T}(\Pi) = \$ a_1 a_2 a_1 b_1 b_2 b_1 a_1 a_2^{i_r} a_1 \dots a_1 a_2^{i_1} a_1$
$\to a)\checkmark b)\checkmark c)\checkmark$ offensichtlich.

IS: IA:
$A_1 \to_G^* a_1 \dots a_k A_{i_1} \dots A_{i_r} \Leftrightarrow (a) \wedge (b) \wedge (c)\checkmark$
sei $\Pi = A_{i_1} \to a_{k+1} A_{j_1} \dots A_{j_t} \in P_a$ dann:
$A_1 \to_G^* a_1 \dots a_{k+1} A_{j_1} \dots A_{j_t} A_{i_2} \dots A_{i_r}$
Induktiv:
$\mu(y_1 \dots y_k) = \$ a_1 a_2^{ir} a_1 \dots a_1 a_2^{i1} a_2$
$\mu(y_1 \dots y_k)\mu(y_{k+1}) = \$ a_1 a_2^{i_r} a_1 \dots \cancel{a_1 a_2^{i_1} a_2 b_1 b_2^{i_1} b_1} a_1 a_2^{i_t} a_1 \dots a_1 a_2^{i_1} a_1$
a),b),c) ✓

Rückrichtung: sind $\mu(y_1 \dots y_k), \mu(y_1 \dots y_{k+1})$
und $A_1 \to_G^* a_1 \dots a_k A_{i_1} \dots A_{i_r}$ wie oben.
So muss eine entgegengesetzte Regel exisitieren womit
$A_1 \to_G^* a_1 \dots a_{k+1} A_{j_1} \dots A_{j_t} A_{i_2} \dots A_{i_r}$ $\square$

2.8 Satz von Parikh

Definition 2.8.1. *$M \subseteq \mathbb{N}^n$ heißt linear g.d.w M ist der Form*
$\{\alpha_0 + n_1\alpha_1 + \dots + n_m\alpha_m | n_i \in \mathbb{N}_0\}$ *für* $\alpha_1 \dots \alpha_m \in \mathbb{N}^n$

Definition 2.8.2. *M heißt semilinear g.d.w.* $\exists$ *lineare Mengen* $M_1 \dots M_k$:
$M = \bigcup_{i=1}^{k} M_i$

Definition 2.8.3 (Parikh Abbildung für Wörter)**.**
$\Psi : \Sigma^* \to \mathbb{N}^n$ *wobei* $\Sigma = \{a_1 \dots a_n\}$ *und* $w \mapsto (\underbrace{|w|_{a_1}}_{Anzahl\ a_1}, |w|_{a_2}, \dots, |w|_{a_n})$
Beispiel: $\Psi(abbaccba) = \Psi(aaabbbcc) = (3, 3, 2)$

Definition 2.8.4 (Parikh Abbildung für Sprachen)**.**
$\Psi(L) = \{\Psi(w) | w \in L\}$
$\Psi(xy) = \Psi(yx) = \Psi(x) + \Psi(y)$
$\Psi(L)$ *heißt Parikhbild von* L

Definition 2.8.5. L_1 *und* L_2 *heißen* zeichenäquivalent *g.d.w* $\Psi(L_1) = \Psi(L_2)$

Satz:

L hat semilineares Parikhbild $\Leftrightarrow$ L ist zeichenäquivalent zu einer reg. Sprache

Beweis:

$\Rightarrow$ $\Psi(L)$ semilinear $\Rightarrow \Psi(L) = M_1 \cup M_2 \cup \dots \cup M_m$
mit $M_i = \{\alpha_{i0} + n_1\alpha_{i1} + \dots n_{ir_i}\alpha_{ir_i} | n_{i,j} \in \mathbb{N}_0, 1 \leq j \leq r_i\}$
seien $y_{ij} \in \Sigma^*$ mit $\Psi(y_{ij}) = \alpha_{ij}$
Grammatik G: $S \to A_1 | \dots | A_m$
$A_i \to A_i y_{i1} | A_i y_{i2} | \dots | A_i y_{ir} | y_{i0}$
es ist $\Psi(L) = \Psi(L(G))$ und $L(G)$ regulär.

$\Leftarrow$ Induktion über reguläre Ausdrücke.

1. $L = L(\emptyset) \Rightarrow \Psi(L) = \emptyset$
2. $L = L(\epsilon) \Rightarrow \Psi(L) = \{0\}$
3. $L = L(a_i) \Rightarrow \Psi(L) = \{(0, \dots, 1, \dots, 0)\}$
4. $L = L_1 \cup L_2$
   $\Psi(L_1), \Psi(L_2)$ semilinear $\Rightarrow \Psi(L) = \Psi(L_1 \cup L_2) = \Psi(L_1) \cup \Psi(L_2)$
5. $L = L_1 L_2$
   $\Psi(L_1), \Psi(L_2) s.l. \Rightarrow \Psi(L) = \Psi(L_1 L_2) = \Psi(L_1) + \Psi(L_2)$
6. $\Psi(L) s.l. \Rightarrow \Psi(L^*) s.l$
   $\Psi(L) = M_1 \cup \dots \cup M_k = \Psi(L_1) \cup \dots \cup \Psi(L_k)$
   $\Psi(L_i^*)$ s.l. da $\Psi(L_i^*)$ und * von linearer Menge semilinear
   $\Psi(L_1^* L_2^* \dots L_k^*) s.l. = \Psi((L_1 \cup L_2 \dots \cup L_k)^*) = \Psi(L^*)$ $\square$

2.8.1 Satz (Parikh 1966)

L k.f $\Rightarrow$ L hat semilineares Prarikhbild
Ohne Beweis.

Konsequenzen

- jede kontextfreie Sprache ist zeichenäquivalent zu einer regulären Sprache
- $L \subseteq \{a\}^*$ k.f. $\Rightarrow$ L reg

2.8.2 Anmerkungen

- Homomorphismus kann mit $h(\$) = \epsilon$ $w\$w^R$ kaputt machen
- nicht-löschender / ϵ-freier Homomorphismus: $h(a) \neq \epsilon \ \forall a \in \Sigma$
- lässt sich trotzdem zerstören: $h(\$) = a, \ a \in \Sigma \setminus \{\$\}$
- $\mathbb{D}_k$ sind VPL
- VPL sind abgeschlossen unter $\cap$
- reguläre Sprachen sind VPL
  - Chomsky-Schützenberger
  - PDA auf $\Sigma \rightsquigarrow$ PDA auf $(\Sigma_c, \Sigma_r, \Sigma_{int})$
    - $h(a_c) = h(a_r) = h(a_{int}) = h(a)$
    - $\Sigma_c = \{a_c \mid a \in \Sigma\}$
    - $\Sigma_r = \{a_r \mid a \in \Sigma\}$
    - $\Sigma_{int} = \{a_{int} \mid a \in \Sigma\}$
- Kellerinhalte einer kontextfreien Sprache sind immer regulär.

3 Algebra

3.1 Wiederholung

3.1.1 Eigenschaften

Betrachte Menge M und Verknüpfung $\circ : M \times M \to M$ ($\circ(m_1, m_2) = m_1 \circ m_2$).
$(M, \circ)$ oder kurz M kann verschiedene Eigenschaften haben:

	assoziativ	neutrales Element	inverse Elemente	kommutativ
Halbgruppe	×			?
Monoid	×	×		?
Gruppe	×	×	×	?
abelsche Gruppe	×	×	×	×

Beispiel

- $(\mathbb{N}, \cdot, 1)$ Monoid
- $(\mathbb{N}, +, 0)$ Monoid
- $(\mathbb{Z}, +, 0)$ Gruppe
- $(2^X, \cap, X)$ kommutatives Monoid
- $(\Sigma^*, \cdot, \epsilon)$ Monoid

3.1.2 Multiplikationstabelle

Sei M endl. Halbgruppe:

	m_1	m_2	$\dots$
m_1	m_1^2	$m_1 m_2$	
m_2	$m_2 m_1$	m_2^2	
$\vdots$			

leicht ablesbar:

- existiert neutrales Element?
- ex. 0-Element?
- ex. inverses Element?
- kommutativ?

Beispiel

$(\mathbb{Z}/3\mathbb{Z}, +) \to (\mathbb{Z}_3, +),\ + : \mathbb{Z}^2 \to \mathbb{Z}_3 : x + y \mapsto (x + y) \mod 3$

	[0]	[1]	[2]
[0]	0	1	2
[1]	1	2	0
[2]	2	0	1

3.1.3 Homomorphismen

Sei $\varphi : G \to H$ und $(G, \cdot), (H, \circ)$ Halbgruppen.
φ ist Homomorphismus $\Leftrightarrow \forall x, y \in G : \varphi(x \cdot y) = \varphi(x) \circ \varphi(y)$ bei Monoiden: $\varphi(e_G) = e_H$
φ heißt:

- injektiv $\Leftrightarrow \forall h \in H : |\varphi^{-1}(h)| \leq 1$ (Monomorphismus, $\varphi : G \hookrightarrow H$)
- surjektiv $\Leftrightarrow \forall h \in H : |\varphi^{-1}(h)| \geq 1$ (Epimorphismus, $\varphi : G \twoheadrightarrow H$)
- bijektiv, $\Leftrightarrow \forall h \in H : |\varphi^{-1}(h)| = 1$ (Isomorphismus, $\varphi : G \cong H$)
- $\varphi : G \to G$ (Endomorphismus)
- bijektiver Endomorphismus heißt Automorphismus

3.1.4 Relationen

$\sim \subseteq X \times X$ ist Relation, $(x_1, x_2) \in \sim \Leftrightarrow: x_1 \sim x_2$
$\sim$ reflexiv, symmetrisch, transitiv: Äquivalenzrelation
$\Rightarrow \sim$ partitioniert X in Äquivalenzklassen $X/\sim$
Sei $(H, \circ)$ eine Halbgruppe und $\sim \subseteq H \times H$. $\sim$ heißt verträglich mit $\circ$, wenn $a \sim b \wedge c \sim d \Rightarrow ac \sim bd$. $\sim$ ist damit eine Kongruenzrelation $\Rightarrow (H/\sim, \circ_\sim)$ ist Halbgruppe.
$\circ : H \times H \to H : a \circ b \mapsto c$
$\circ_\sim : H/\sim \times H/\sim \to H : [a]_\sim \circ_\sim [b]_\sim \mapsto [c]_\sim$

Normalteiler

$N < G$: N ist Untergruppe von G
$N \lhd G$: N ist Normalteiler von G, d.h. $\forall g \in G : gN = Ng$ ($\Leftrightarrow \forall g \in G : gNg^{-1} = N$)
Normalteiler und Kongruenzrelation stehen in 1-zu-1-Zusammenhang: Nebenklassen $\leftrightarrow$ Äquivalenzklassen; z.B. $\mathbb{Z}/n\mathbb{Z}$

3.1.5 Freies Monoid

(freie Gruppe, ...)
Zwei äquivalente Definitionen:

1. M heißt frei über $A \subseteq M \Leftrightarrow \forall m \in M : m$ ist eindeutiges Produkt von Elementen aus A (z.B. $\Sigma \subseteq \Sigma^*$)

2. M heißt frei über $A \subseteq M \Leftrightarrow \exists g : A \hookrightarrow M$ und $f : A \to X$ beliebig (X Monoid) $\Rightarrow \exists! \varphi : M \to X$

Beispiel

- $(\Sigma^*, \cdot)$ ist frei über Σ
- $(\mathbb{N}, +)$ ist frei über $\{1\}$
- jede Gruppe/Monoid ist Faktorgruppe/-monoid einer freien Gruppe $\rightarrow$ frei heißt frei von Relationen
- $\left(\left(\Sigma \cup \bar{\Sigma}\right)^*, \cdot\right) /_{a\bar{a}=1} \cong F(\Sigma)$ (freie Gruppe über Σ)
- $F(\Sigma)/_{xy=yx}$ ist freie abelsche Gruppe

3.2 Spracherkennung durch Monoide

Sei $L \subseteq \Sigma^*$, M Monoid und $\varphi : \Sigma^* \to M$ Homomorphismus. L wird von M erkannt (mit φ) $\Leftrightarrow \varphi^{-1}(\varphi(L)) = L$
äquivalent: $\exists X \subseteq M : \varphi^{-1}(X) = L$

Beispiele

- $L = \Sigma^*$ wird von $\{1\}$ mit $\varphi : x \mapsto 1$ erkannt.
- $L = \{w \in \Sigma \mid |w| \equiv 0 \pmod 2\}$ wird von $\mathbb{Z}/2\mathbb{Z}$ erkannt
  - $\varphi(a) = [1]$, also $L = \varphi^1([0])$
  - $\varphi(\epsilon) = [0]$
  - das ginge mit jedem $\mathbb{Z}/n\mathbb{Z}$ mit n gerade
- Σ^* erkennt L beliebig mit $\varphi = id$
  - unendliche Monoide können "zu viel"
  - Konzept nur für endliche Monoide interessant

  $\rightarrow$ Ziel: für Sprache L kleinstes erkennendes Monoid finden

3.2.1 Das syntaktische Monoid

Kongruenzrelation

Myhill-Nerode-Äquivalenz:

- $x \sim_L^L y \Leftrightarrow \forall z : xz \in L \Leftrightarrow yz \in L$
- $x \sim_L^R y \Leftrightarrow \forall z : zx \in L \Leftrightarrow zy \in L$

$\rightarrow \sim_L^L,\ \sim_L^R$ sind keine Kongruenzrelation

für Kongruenzrelation muss gelten:

$$u \sim v \wedge w \sim x \Rightarrow uw \sim vx$$

Beispiel:

- $L = ac^+bc^+$
- Wähle $u = v = a,\ w = b,\ x = c$
- $uw = ab \not\sim vx = ac$

Syntaktische Kongruenz

$x \sim_L y :\Leftrightarrow \forall u, v \in \Sigma^* : uxv \in L \Leftrightarrow uyv \in L$ es ist $x \sim_L y \Leftrightarrow x \sim_L^L y \wedge x \sim_L^R y$

Syntaktisches Monoid

Sei $L \subseteq \Sigma^*$, dann ist $Synt(L) := \Sigma^*/\sim_L$ das syntaktische Monoid
$\eta_L : \Sigma^* \rightarrow Synt(L),\ w \mapsto [w]$ heißt syntaktischer Morphismus

Satz: L reg $\Leftrightarrow$ $Synt(L)$ endlich

Satz: $Synt(L)$ erkennt L mittels η_L

Def.: $N \prec M :\Leftrightarrow \exists M' \leq M \exists \psi : M' \twoheadrightarrow N$. "$N$ teilt M"

Satz: M erkennt $L \Leftrightarrow Synt(L) \prec M$

1) Falls M endlich, so gilt:
   $\varphi : M \twoheadrightarrow M \Leftrightarrow \varphi$ Monomorphismus
2) $Synt(L)$ erkennt L
3) $\prec$ ist transitiv
4) $Synt(L_1 \cap L_2) \prec Synt(L_1) \times Synt(L_2)$

4 Reguläre Sprachen

4.0.1 Darstellungen

DEA, NEA, reg. Ausdrücke, endl. Monoide, Typ-3-Grammatik, Logik

4.1 Definition: Transformationsmonoid

Sei M ein DEA, Q Zustände, $w \in \Sigma^*$
$f_w : Q \to Q : q \mapsto \delta^*(q, w)$
$Trans(M) := \{f_w \mid w \in \Sigma^*\}$

4.1.1 Lemma

$Trans(M)$ ist endl. Monoid mit:

- $f_\epsilon = id_Q$ neutrales Element
- $f_x \cdot f_y = f_{xy}$
- $|Trans(M)| \leq |Q|^{|Q|}$

4.2 Satz

L reg. $\Leftrightarrow$ L wird von endlichem Monoid H erkannt

4.2.1 Beweis

$\Rightarrow$: $Trans(M)$ erkennt L mit $\varphi : \Sigma^* \to Trans(M) : w \mapsto f_w$,
$X = \{f : Q \to Q \mid f(q_0) \in F\}$
$w \in L \Rightarrow \exists$ Lauf von q_0 nach $f \in F \Leftrightarrow f_w(q_0) \in F \Leftrightarrow \varphi(w) \in X$

$\Leftarrow$: Konstruiere M aus H:
$M = (Q, \Sigma, \delta, q_0, F), Q = H, q_0 = 1, F = X, \delta(q, a) = q \cdot \varphi(a)$ $\square$

4.3 Satz

DFA M minimal $\Leftrightarrow Synt(L(M)) \simeq Trans(M)$

4.3.1 Beweis

$\Leftarrow$: ergibt sich aus Satz oben ($Trans(M)$ kann nicht kleiner sein als das $Synt(L(M))$)

$\Rightarrow$: z.z. M min $\Rightarrow (f_x = f_y \Leftrightarrow x \sim_L y)$

$\Rightarrow$: Folgt aus Satz

$\Leftarrow$: ang. $x \sim_L y$ aber $f_x \neq f_y$
also $\exists u, v : q_1 = f_{uxv}(q_0) \neq f_{uyv}(q_0) = q_2$
also $\forall v' : f_{v'}(q_1) \in F \Leftrightarrow f_{v'}(q_2) \in F$
dann aber Automat nicht minimal $\lightning$ □

$\Rightarrow$ Syntaktisches Monoid aus Minimalautomat berechenbar.

5 Logik

Wir betrachten logische Formeln, die auf Wörtern operieren.
Sei $w \in \Sigma^*$ und φ logische Formel:
$w \models \varphi$, wenn w Modell für φ, also φ wahr unter w.

5.1 Beispiel

- $\varphi_1 = \exists x \exists y (\underbrace{\forall z(z \geq x)}_{x=1} \wedge Q_a x \wedge \underbrace{\forall z(z \leq y)}_{y=|w|} \wedge Q_b y),\ x, y, z \in \{1, \ldots, |w|\}$
  - $L(\varphi_1) = a\Sigma^* b$
  - Quantifizierung erster Ordnung
  - Variablen: Positionen im Wort
  - Prädikate: $<,\ Q_a,\ Q_b$; $Q_a x$ wahr $\Leftrightarrow$ a an Stelle x
  - alle Variablen müssen gebunden sein
- $\varphi_2 = \exists X \forall x\, (\forall z(z \geq x) \Rightarrow X(x)) \wedge \forall x\, (\forall z(z \leq x) \Rightarrow \neg X(x)) \wedge$
  $\forall x \forall y\, (((x < y) \wedge \forall z(z > x \Rightarrow z \geq y)) \Rightarrow (X(x) \Leftrightarrow \neg X(y)))$
  - $L(\varphi_2) = \left(\Sigma^2\right)^*$
  - $\exists X$ ist Quantor zweiter Ordnung
  - X ist einstelliges Prädikat, d.h. monadisch
  - X beschreibt Menge von Positionen im Wort
  - erste Zeile: erste Position im Wort in X, letzte nicht
  - zweite Zeile: benachbarte Position $\Rightarrow$ genau eine in X

5.1.1 Anmerkungen

- $x = y \equiv x \leq y \wedge y \leq x$
- $x < y \equiv x \leq y \wedge \neg(x = y)$
- $x + 1 = y \equiv x < y \wedge \forall z(z > x \Rightarrow z \geq y)$
- aber $\leq$ nicht durch $+1$ simulierbar in FO (aber $MSO[+1] = MSO[<]$)
- auch definierbar: $first(x),\ last(x),\ X \subseteq Y$, uvm.

5.2 Definition

- atomare Formeln sind Prädikate $R_i^j \subseteq D^j$
- D ist Domäne, also Menge aller Wortpositionen
- $R_i^j(x_1, \ldots, x_j)$ ist atomare Formel (mit freien Variablen)
- Sind φ_1, φ_2 Formeln, so auch $\varphi_1 \wedge \varphi_2$ und $\neg\varphi_1$
- Ist φ Formel, so auch $\exists x\varphi$ und $\exists X\varphi$

5.3 Notation

Die Notation ist abhängig davon, welche "Bausteine" eine Formel verwendet. Beispielsweise: $FO[<]$: Quantifizierung erster Ordnung und $<$-Prädikat (Q_a Quantor immer gegeben)

5.3.1 Anmerkungen

- $FO[+1] \subsetneq FO[<]$
- $MSO[+1] = MSO[<] = MSO$ "monadic second order"
- $FO + MOD[<]$: MOD_k : Durch k teilbare Zahl (von 'a's) $MOD_k x Q_a x$
- $FO[+] : x + y = z$

5.4 Logik auf Worten

5.4.1 Semantik

Definition

Sei Σ ein Alphabet, V_1 eine Menge von FO-Variablen und V_2 von MSO-Variablen.
Eine V_1V_2-Struktur ist ein Wort über $\Sigma \times \mathcal{P}(V_1) \times \mathcal{P}(V_2)$,
d.h. $w = (a_1, X_1, Y_1), (a_2, X_2, Y_2), \ldots, (a_n, X_n, Y_n)$
mit:

1. $X_i \cap X_j = \emptyset$ falls $i \neq j$
2. $\bigcup_{i=1}^{n} X_i = V_1$

Definition

Sei φ eine MSO-Formel in der keine Variable mehrfach gebunden wird mit freien Variablen $V_1,\ V_2$.
Die Modellrelation $\models$ zwischen φ und (V_1, V_2)-Strukturen ist induktiv definiert. Im Folgenden sei $w = (a_1, X_1, Y_1), (a_2, X_2, Y_2), \ldots, (a_n, X_n, Y_n)$ eine (V_1, V_2)-Struktur.

- $w \models Q_a x$, falls es ein $1 \leq i \leq n$ gibt mit $a_i = a$ und $x \in X_i$
- $w \models X(x)$, falls es ein $1 \leq i \leq n$ gibt mit $x \in X_i$ und $X \in Y_i$
- $w \models P(x_1, \ldots, x_m)$ für ein m-äres Prädikat, falls es $i_1, \ldots, i_m$ gibt mit $x_j \in X_{i_j}$ und $(i_1, \ldots, i_m) \in P$
- $w \models \neg\psi$ falls $w \not\models \psi$
- $w \models \psi_1 \wedge \psi_2$ falls $w \models \psi_1 \wedge w \models \psi_2$
- $w \models \exists x : \psi$ falls ein $1 \leq i \leq n$, sodass die $(V_1 \cup \{x\}, V_2)$-Struktur

  $$w' = (a_1, X_1, Y_1) \ldots (a_i, X_i \cup \{x\}, Y_i) \ldots (a_n, X_n, Y_n)$$

  $w' \models \psi$ erfüllt.
- $w \models \exists X : \psi$, falls es eine Menge $M \subseteq \{1, \ldots, n\}$ gibt, sodass $(V_1, V_2 \cup \{X\})$-Struktur
  $w' = (a_1, X_1, Y_1) \ldots (a_n, X_n, Y_n)$ mit $Y_i = \begin{cases} Y_i & \text{falls } i \notin M \\ Y_i \cup \{X\} & \text{sonst} \end{cases}$

Bemerkung

Falls wir uns mit einer FO-Formel befassen, verwenden wir V-Strukturen, d.h. Worte über $\Sigma \times \mathcal{P}(V)$. MSO-Formeln ohne freie Variablen (MSO-Sätze) besitzen also $(\emptyset, \emptyset)$-Strukturen, also Wörter, als Modelle.

Definition

Sei φ ein MSO-Satz. Die von φ definierte Sprache ist $L_\varphi = \{w \in \Sigma^* \mid w \models \varphi\}$

5.5 Rückblick (FS1)

5.5.1 Ergebnisse

Es wurde gezeigt:

- $FO[<] = SF$ (Sternfreie Sprachen, Siehe auch UB 7)
- $MSO[+1] = REG$

Beispiel

- $FO[<]$:
  - $\forall x : (Q_a x \Rightarrow \exists y : Q_b y \wedge x < y)$
  - $\forall x : Q_a x \Rightarrow \forall y : Q_b y \ (\Sigma^* \setminus a^*)$
  - $\exists x \forall y : x < y \Rightarrow Q_a y \ (\Sigma^*)$
- $MSO[<] = MSO[+1]$
  - $x < y \equiv \exists Z : (y \in Z \wedge x \notin Z \wedge \forall z : (z \in Z \Rightarrow \exists w : w \in Z \wedge z + 1 = w))$
  - $x \leq y \equiv \forall z : z < x \Rightarrow z < y$
  - $fst(x) \equiv \forall y : x \leq y$, $lst(x) \equiv \forall y : y \leq x$
  - $\exists M \exists x \, (fst(x) \wedge x \in M \wedge \forall z : (z \in M \Leftrightarrow (\forall w : (w = z + 1 \Rightarrow w \notin M)))) \wedge \exists y (lst(y) \wedge y \notin M)$ (gerade Länge)

6 Satz von Schützenberger

6.1 Satz

Eine Sprache L ist genau dann sternfrei, wenn L von einem aperiodischen Monoid erkannt wird (bzw. $Synt(L)$ ist aperiodisch).

6.2 Definition

Sei M Monoid. $R \subseteq M$ ist

- Rechts-ideal, falls $RM = R$
- Links-ideal, falls $MR = R$
- Ideal, falls $MRM = R$

6.2.1 Beispiel

- für jedes Monoid M sind M und $\emptyset$ Ideale
- für $M = \Sigma^*$ mit $a \in \Sigma^*$ ist $\Sigma^* a \Sigma^*$ ein Ideal, $a\Sigma^*$ ein Rechtsideal und $\Sigma^* a$ ein Linksideal.

6.3 Definition: Green'sche Relationen

Wir definieren die sogenannten Green'schen Relationen auf M via:

- $a \leq_R b \Leftrightarrow aM \subseteq bM$
- $a \leq_L b \Leftrightarrow Ma \subseteq Mb$
- $aIb \Leftrightarrow MaM = MbM$
- $aRb \Leftrightarrow aM = bM$
- $aLb \Leftrightarrow Ma = Mb$

6.3.1 Lemma

Sei M aperiodisch und $p, q, r \in M$ mit $pqr = q$, dann ist auch $pq = qr = q$ (Bew. ÜB)

6.3.2 Lemma

Sei M aperiodisch und $a, b \in M$ mit aIb. Dann gilt:

1. $a \leq_R b \Rightarrow aRb$
2. $a \leq_L b \Rightarrow aLb$

Beweis

Da $MaM = MbM$ gilt, gibt es $u, v \in M$ mit $b = uav$.

1. Wegen $aM \subseteq bM$ gibt es $p \in M$ mit $a = bp$. Also $b = uav = ubpv$, nach Lemma 6.3.1 ist also $ub = b$. Damit ist $b = bpv$, also $bM = bpvM \subseteq bpM \subseteq aM$. Also $bM = aM$, somit aRb
2. symmetrisch

6.3.3 Lemma

Sei M aperiodisches Monoid und $p \in M$. Dann ist $\{p\} = (Mp \cap pM) \setminus J_p$ mit $J_p = \{s \in M \mid p \notin MsM\}$

Beweis

$\subseteq$ klar $(1 \in M)$

$\supseteq$ Sei $x \in (Mp \cap pM) \setminus J_p$.
Dann gibt es $q, r \in M$ mit $pq = x = rp$. Da $x \notin J_p$, ist $p = uxv$ für $u, v \in M$.
Damit:

$$p = uxv = (ur)p(v) = urp = ux = upq = pq = x \Rightarrow x \in \{p\}$$

□

6.4 Beweis zum Satz von Schützenberger

Zu zeigen: ist $Synt(L)$ aperiodisch, so ist L sternfrei.
$Synt(L)$ endlich, da $L \in$ Reg Sei $\varphi : \Sigma^* \to M = Synt(L)$ ein Monomorphismus mit M aperiodisch. Da φ L erkennt, existiert eine endliche Teilmenge $P \subseteq M$ mit $\varphi^{-1}(P) = \bigcup_{m \in P} \varphi^{-1}(m) = L$. Wir zeigen $\varphi^{-1}(\{m\})$ ist sternfrei $\forall m \in M$:
Induktion über $r(m) = |M \setminus MmM|$:

$r(m) = 0$: Dann ist $M = MmM$ (Differenz leer, abgeschlossen). Damit gibt es $p, q \in M$ mit $1 = pmq = pm1q = pm = p1m = m$ (nach Lemma 6.3.1).

Folglich: $\varphi^{-1}(m) = \varphi^{-1}(1) \overset{(*)}{=} \{a \in \Sigma \mid \varphi(a) = 1\}^*$, denn ist $w \in \Sigma^*$ mit $|w| > 1$ und $w = ua,\ a \in \Sigma$, dann ist

$$\varphi(w) = \varphi(ua) = 1 \Leftrightarrow \varphi(u)\varphi(a) = 1 \Leftrightarrow \varphi(u)1\varphi(a) = 1 \Leftrightarrow \varphi(u) = 1 \wedge \varphi(a) = 1$$

$(*)$ folgt durch Induktion.
Ist $B \subseteq \Sigma$, so ist B^* sternfrei: $B^* = \overline{\overline{\emptyset}\overline{B}\overline{\emptyset}}$. Damit ist $\varphi^{-1}(m)$ sternfrei.

$r(m) > 0$: Wir zeigen: $\varphi^{-1}(m) = (U\Sigma^* \cap \Sigma^*V) \setminus (\Sigma^*C\Sigma^* \cup \Sigma^*W\Sigma^*) = K$ mit:

- $U = \bigcup_{(n,a)\in E} \varphi^{-1}(n)a\Sigma^*$
- $V = \bigcup_{(n,a)\in F} \Sigma^*a\varphi^{-1}(n)$
- $C = \{a \in \Sigma \mid m \notin M\varphi(a)M\}$
- $W = \bigcup_{(a,n,b)\in G} \Sigma^*a\varphi^{-1}(n)b\Sigma^*$
- $E = \{(n,a) \in M \times \Sigma \mid n\varphi(a)Rm,\ n \notin mM\}$
- $F = \{(n,a) \in M \times \Sigma \mid \varphi(a)nLm,\ n \notin Mm\}$
- $G = \{(a,n,b) \in \Sigma \times M \times \Sigma \mid m \in (M\varphi(a)nM \cap Mn\varphi(b)M) \setminus M\varphi(a)n\varphi(b)M\}$

Dann ist nur noch zu zeigen, dass U, V, W, C sternfrei.

$\varphi^{-1}(m) \subseteq K$: Wir nehmen an $m \neq 1$ (sonst $r(m) = 0$). Sei $u \in \varphi^{-1}(m)$ und $p \in \Sigma^*$ sei das kürzeste Präfix von u mit $\varphi(p)Rm$.
Da $\varphi(u) = m$ ist, muss es ein solches p geben.

- Falls $p = \epsilon$, ist $1Rm$ also $M = mM$ und damit $m = 1$ ↯ $\Rightarrow p \neq \epsilon$
- Falls $p = ra,\ a \in \Sigma,\ r \in \Sigma^*$:
  Setze $n = \varphi(r)$. Dann ist $\varphi(p) = n\varphi(a)Rm$ aber nicht nRm also $(n,a) \in E$ und somit $u \in U\Sigma^*$
- Analog für $u \in \Sigma^*V$

Es bleibt zu zeigen, dass $u \notin \Sigma^*C\Sigma^* \cup \Sigma^*W\Sigma^*$.

- Angenommen $u \in \Sigma^*C\Sigma^*$, dann ist $u = sct$ mit $s,t \in \Sigma^*,\ c \in C$. Aber: $m = \varphi(u) \in M\varphi(c)M$ im Widerspruch zu $c \in C \Rightarrow u \notin \Sigma^*C\Sigma^*$
- Angenommen $u \in \Sigma^*W\Sigma^*$, dann ist $u \in \Sigma^*a\varphi^{-1}(n)b\Sigma^*$ für ein $(a,n,b) \in G$.
  Damit ist $m = \varphi(u) \in M\varphi(a)n\varphi(b)M$ im Widerspruch zu $(a,n,b) \in G$ ↯ $\Rightarrow u \notin \Sigma^*W\Sigma^*$

$\Rightarrow \varphi^{-1}(m) \subseteq K$

$K \subseteq \varphi^{-1}(m)$: Sei $u \in K$ und $s = \varphi(U)$. Da $u \in U\Sigma^*$, ist $s \in n\varphi(s)M$ für ein $(n,a) \in E$.
Also $n\varphi(a)Rm$, damit $s \in mM$.
Analog $u \in \Sigma^*V \Rightarrow s \in Mm$.
Also $s \in mM \cap Mm$.

Es gilt: $m \in MsM \Leftrightarrow s \notin J_m$ (Definition in Lemma 6.3.3) und mit dem Lemma folgt $s \in (mM \cap Mm) \setminus J_m = \{m\}$
Also genügt zu zeigen $m \in MsM$.
Angenommen $m \notin MsM$.
Dann gibt es $f \in \Sigma^*$ minimal mit $s = vfw$ und $m \notin M\varphi(f)M,\ v, w \in \Sigma^*$

* Falls $f = \epsilon \Rightarrow m \notin M$ ↯
* Falls $f = a, a \in \Sigma \Rightarrow a \in C \Rightarrow u \in \Sigma^*C\Sigma^* \Rightarrow u \notin K$ ↯
* Falls $f = agb$ mit $a, b \in \Sigma,\ g \in \Sigma^*$; setze $x = \varphi(g)$. Dann ist $\varphi(f) = \varphi(a)x\varphi(b)$ und $u \in M\varphi(a)xM \cap Mx\varphi(b)M$, da f minimal. Also $(a, x, b) \in G$, damit $f \in W$ und $u \in \Sigma^*W\Sigma^* \Rightarrow u \notin K$ ↯

$\Rightarrow m \in MsM$, letztlich $K \subseteq \varphi^{-1}(m)$

– U, V, W, C sternfrei

* C sternfrei, klar da endlich
* zu U (V analog):
  $(n, a) \in E \Rightarrow n\varphi(a)Rm \Rightarrow Mn\varphi(a)M = MmM \subseteq MnM$
  $\Rightarrow r(n) \leq r(m)$
  - Ist $r(n) = r(m)$, so $MmM = MnM$, damit mIn. Da $mM = n\varphi(a)M \subseteq nM$, also $m \leq_R n$, folgt mit Lemma 6.3.2 nRm, insbesondere $n \in mM$ ↯
  - Also $r(n) < r(m)$
    IV: $\varphi^{-1}(n)$ ist sternfrei.
    $\Rightarrow U$ ist sternfrei.
* zu W:
  Sei $(a, n, b) \in G$. Dann ist $m \in M\varphi(a)nM \subseteq MnM$
  $r(n) \leq r(m)$
  - Angenommen $r(n) = r(m)$. Dann ist $MnM = MmM$ also insbesondere $n \in MmM$. Da $m \in Mn\varphi(b)M$ ist dann auch $n \in Mn\varphi(b)M$ also $n = rn\varphi(b)s = n\varphi(b)s,\ r, s \in M$.
    Analog $m \in M\varphi(a)nM \Rightarrow n = x\varphi(a)ny \Rightarrow n = x\varphi(a)n\varphi(b)sy \in M\varphi(a)n\varphi(b)M$ ↯ (Widerspruch zu $(a, n, b) \in G$)

  $\Rightarrow r(n) < r(m)$

  $IV : \varphi^{-1}(n)$ sternfrei, damit W sternfrei □

7 Schaltkreiskomplexität

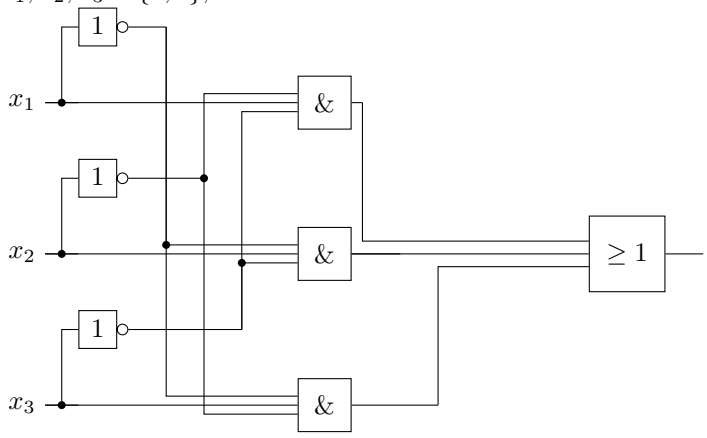
7.1 Schaltkreise

Schaltkreise bestehen aus:

- Eingangsgatter
- Ausgänge
- Gatter
- Verbindungen (DAG)

7.1.1 Beispiel

$x_1, x_2, x_3 \in \{0,1\}$, $L = 0^*10^*$



7.1.2 Anmerkungen, Schaltkreisfamilien

Mögliche Gatter: AND, OR, MOD, MAJ, ...
Für Spracherkennung: Schaltkreise für verschiedene Eingabelängen $\rightarrow$ Schaltkreisfamilien $(C_n)_{n \in \mathbb{N}}$
Bsp. leicht zu Familie erweiterbar.

Problematik

Schaltkreisfamilien können beispielsweise unäres Halteproblem entscheiden
→ Uniformität: Es existiert Algorithmus, der auf Eingabe n den Schaltkreis C_n berechnet. (z.B. $DLOGTIME$-uniform)

7.2 Schaltkreise als Alternative zu Turingmaschinen

Komplexitätsmaße bei TM: Zeit, Platz Komplexitätsmaße bei Schaltkreisen: Größe, Tiefe, Gatter, Fan-in, Uniformität

7.2.1 Klassen

NC

NC^i: polynomiell viele Gatter, $\mathcal{O}\left(\log^i(n)\right)$ Tiefe, Fan-in=2, Bool'sche Gatter
NC: $\bigcup_i NC^i$

AC

AC^i: wie NC^i nur unbeschränkter Fan-in
AC : $\bigcup_i AC^i$

ACC

ACC^i_k: Wie AC^i nur mit MOD_k-Gatter zusätzlich
ACC^i: $\bigcup_k ACC^i_k$

TC

TC^i: wie AC^i, ausschließlich MAJ-Gatter
TC : $\bigcup_i TC^i$

SAC

SAC^i: wie AC^i nur $ODER$-Gatter haben unbeschränkten Fan-in
SAC: $\bigcup_i SAC^i$

CC

CC^i: Wie AC^i, ausschließlich MOD-Gatter
CC : $\bigcup_i CC^i$

Anmerkung

- Besonders interessant: $AC^0,\ ACC^0,\ TC^0,\ NC^1$
- $NC = AC = ACC = TC = SAC$
- $AC^{i-1} \subseteq NC^i$ (offensichtlich: Fan-in durch Tiefe)

7.2.2 Satz

PARITY nicht in AC^0 (Furst, Saxe, Sipser).

Anmerkung

Also $AC^0 \subsetneq NC^1$.
Offen:

- $TC^0 \overset{?}{\subsetneq} NC^1$
- $ACC^0 \overset{?}{\subsetneq} TC^0$

7.2.3 Satz

Reguläre Sprachen (REG) $\subseteq NC^1$

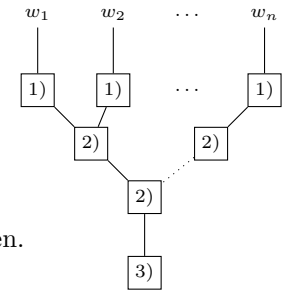
Beweis

Sei L regulär, oBdA $L \subseteq \{0,1\}^*$
Sei $|Synt(L)| = k$ und $\eta_L : \Sigma^* \to Synt(L) : w \mapsto [w]$
nummeriere alle $m \in Synt(L)$ binär ($\lceil \log k \rceil$ Stellen): $\widehat{m}$
Wir benötigen folgende Schaltkreise/Gatter:

1) $a \in \Sigma :\ a \mapsto \widehat{\eta_L(a)}$

2) auf Eingabe $\widehat{m_1},\ \widehat{m_2}$, gib $\widehat{m_1 m_2}$ aus

3) gib 1 aus gdw. für Eingabe $\widehat{m}$ gilt $m \in \eta_L(L)$



$\to$ Größe von 1),2),3) konstant in Wortlänge
mit Baum log-Tiefe kann Schaltkreisfamilie für L angegeben werden.

7.2.4 Satz

REG ist NC^1 vollständig (beweisen wir hier nicht)

7.2.5 Satz von Buss

Auswertungsproblem für Bool'sche Formeln ist NC^1-vollständig.

7.2.6 Satz

$VPL \subseteq NC^1$ (Dymond "input-driven PDA") und NC^1-vollständig, da Reg$\subseteq VPL$

7.2.7 Satz

CFL ist SAC^1-vollständig.

7.2.8 Satz

$FO[arb] = AC^0$

Beweis $\subseteq$

$\exists x \varphi(x) \to \bigvee_{i=1}^{n} \varphi(i)$
$\forall x \varphi(x) \to \bigwedge_{i=1}^{n} \varphi(i)$
$Q_0 i,\ Q_1 i \to$ Eingang x_i bzw. $\bar{x}_i$
$R(i_1, \ldots, i_k) \to 0$ bzw. 1, je nach Wert von $(i_1, \ldots, i_k)$

7.2.9 weitere Sätze

- $MOD_q[arb] = CC_q^0$
- $FO + MOD_q[arb] = ACC_q^0$
- $DLOGTIME$-uniform $AC^0 = FO[+, \times]$

8 ω-Sprachen

8.1 Einführung

8.1.1 Sprachen bisher

Teilmenge von $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \ldots$
Ein Wort aus Σ^n kann als Funktion $\{1, \ldots, n\} \to \Sigma$ aufgefasst werden.

8.1.2 unendliche Sprachen

Verallgemeinerung auf $\mathbb{N} \to \Sigma$
Menge unendlicher Wörter über Σ: Σ^ω
$\Sigma^\infty := \Sigma^\omega \cup \Sigma^*$

Beispiel

$\{ab\}^*\{bba\}^\omega \cup \{aa\}^\omega$

- Es gibt kein letztes Zeichen
- Konkatenation ist eingeschränkt

8.2 Definition: ω-reguläre Sprachen

- A^ω ist ω-regulär, wenn $A \neq \emptyset$ und regulär
- AB ist ω-regulär, wenn A regulär und B ω-regulär
- $A \cup B$ ist ω-regulär, wenn A und B ω-regulär

8.2.1 Beispiel: Komplementbildung

$(\{0,1\}^*\{0\}^\omega)^c = \{0,1\}^*\{1\}^\omega \cup (\{0,1\}^* \setminus 0^*)^\omega$

8.3 ω-Automaten

8.3.1 Problem

Für endliche Wörter akzeptiert ein Automat, wenn er *nach* dem letzten Zeichen in akzeptierendem Zustand ist. Das geht im ω-Fall nicht (kein letztes Zeichen).

8.3.2 Definition

$\mathcal{A} = (Q, \Sigma, q_0, \delta, Acc)$

- Q Zustände
- Σ Alphabet
- $q_0 \in Q$ Startzustand
- $\delta \subseteq Q \times \Sigma \times Q$ (nicht deterministisch)
  $\delta : Q \times \Sigma \to Q$ (deterministisch)
- Acc Akzeptanzbedingung, je nach Automatentyp

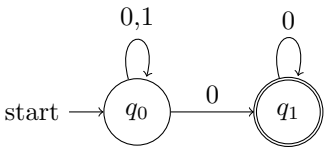
8.3.3 Definition

Sei $x \in \Sigma^\omega$, dann ist $\rho \in Q^\omega$ ein Lauf auf $x \Leftrightarrow (\rho_i, x_i, \rho_{i+1}) \in \delta$ bzw. $\rho_{i+1} = \delta(\rho_i, x_i)$
$In(\rho) := \{q \in Q \mid \exists^\omega i : \rho_i = q\}$ "es existieren unendlich viele"

8.3.4 Akzeptanzbedingungen

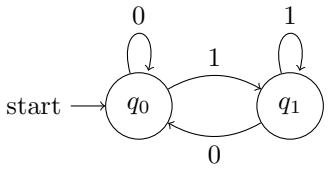
Büchi

$Acc = F \subseteq Q$
$x \in L(\mathcal{A}) \Leftrightarrow In(\rho) \cap F \neq \emptyset$



Muller

$Acc = \mathcal{F} \subseteq 2^Q$
$x \in L(\mathcal{A}) \Leftrightarrow \bigvee_{F \in \mathcal{F}} In(\rho) = F$



$\mathcal{F} = \{\{q_0\}, \{q_0, q_1\}\}$

Rabin

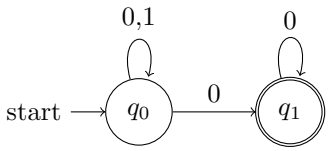
$Acc = \Omega \in (2^Q \times 2^Q)^*$
$\Omega = \{(E_1, F_1), (E_2, F_2), \ldots, (E_n, F_n)\}$
$x \in L(\mathcal{A}) \Leftrightarrow \bigvee_{i=1}^{n} (In(\rho) \cap E_i = \emptyset \wedge In(\rho) \cap F_i \neq \emptyset)$

Streett

Acc wie bei Rabin, aber:
$x \in L(\mathcal{A}) \Leftrightarrow \bigwedge_{i=1}^{n} (In(\rho) \cap E_i \neq \emptyset \vee In(\rho) \cap F_i = \emptyset)$

8.3.5 Satz

Folgende Modelle sind gleichmächtig: Muller,Rabin,Streett, nicht-deterministischer Büchi
Deterministischer Büchi ist echt schwächer. Trennbeispiel:



nicht determinisierbar

8.3.6 Satz

nicht-deterministischer Büchi $\approx \omega$-reguläre Sprachen

8.3.7 Komplementabschluss

Ist nicht-deterministischer Büchi unter Komplement abgeschlossen?
Für deterministische Automaten klar:

- Büchi $F \to Q \setminus F$? Nein!
- Muller $\mathcal{F} \to 2^Q \setminus \mathcal{F}\checkmark$

8.3.8 Satz

Es existiert eine Umwandlung vom nicht-deterministischen Büchi zum deterministischen Muller-Automaten.

8.3.9 Satz

ω-reguläre Sprachen sind unter Komplement abgeschlossen (klar, s.o.)

8.4 Logik für ω-reguläre Sprachen

MSO (und FO) Formeln können auf ω-Wörtern interpretiert werden.
Zum Beispiel: $\forall x \exists z(x < z \wedge Q_a z)$ "unendlich viele a's"

8.4.1 Satz

$MSO = \omega$-regulär

8.5 LTL - Linear Time Logic

- oft kompakter als MSO bzw. FO
- wird in Praxis verwendet

8.5.1 Definition

LTL-Syntax

- $a \in \Sigma$ sind atomare Formeln
- sind φ und ψ LTL-Formeln, so auch $\neg\varphi,\ \varphi \wedge \psi,\ X\varphi,\ G\varphi,\ F\varphi,\ \psi U\varphi$
- auch auftretende Schreibweise:
  - $X\varphi \to \circ\varphi$
  - $G\varphi \to \Box\varphi$
  - $F\varphi \to \diamond\varphi$

LTL-Semantik

- $x \models a :\Leftrightarrow x_1 = a$
- $x \models \neg\varphi :\Leftrightarrow x \not\models \varphi$
- $x \models \varphi \wedge \psi :\Leftrightarrow x \models \varphi \wedge x \models \psi$
- $x \models X\varphi :\Leftrightarrow x_2 x_3 \ldots \models \varphi$
- $x \models G\varphi :\Leftrightarrow x \models \neg F \neg\varphi$
- $x \models F\varphi :\Leftrightarrow \exists i : x_i x_{i+1} \ldots \models \varphi$
- $x \models \varphi U \psi :\Leftrightarrow \exists i \forall k < i : x_i x_{i+1} \ldots \models \psi \wedge x_k x_{k+1} \ldots \models \varphi$ "until"

8.5.2 Satz

LTL$=$ $FO[<]$

8.6 Anwendungsbeispiel:: Model-Checking

8.6.1 Idee

- Erfüllt ein System eine Eigenschaft?
- System: endlich, nicht-terminierend
- Kripke-Struktur $\rightarrow$ Büchi-Automat
- Eigenschaft: z.B. LTL-Formel φ
- Aufgabe: gilt $L(\mathcal{A}) \subseteq L(\varphi)$
  $\Leftrightarrow L(\mathcal{A}) \cap \widehat{L(\varphi)} = \emptyset$
  $\Leftrightarrow \underbrace{L(\mathcal{A}) \cap L(\neg\varphi)}_{\text{Büchi-Automat}} = \emptyset$
- $\rightarrow$ Leerheitstest für Büchi-Automaten:
  1. Finde von Startzustand erreichbaren Endzustand ($\rightarrow$ Graphenerreichbarkeit)
  2. gibt es Endzustandsschleife?

9 Baum-Sprachen

9.1 Einführung

Bisher: Wörter, d.h. gerichtete, zusammenhängende Grad1-Graphen
oder: Funktion $\{1, \ldots, n\} \to \Sigma$

jetzt: Bäume

9.1.1 Bäume

Was ist ein Baum?
Graph mit folgenden Eigenschaften:

- azyklisch
- zusammenhängend (sonst "Wald")
- gerichtet
- gelabelte Knoten
- Rang (1 - Wörter, 2 - Binärbäume,...)
  je nach Definition:
  - label-abhängig
  - ohne Rang
- endlich oder unendlich

Wir betrachten: endliche Binärbäume

9.1.2 Baumsprachen

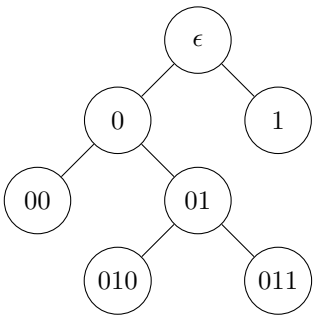
Baumsprache $L \subseteq T_\Sigma^*$

9.2 Knotenadressierung

- bei Wörtern: $\{0, \ldots, n\} \to \Sigma$
- bei Bäumen: $A \to \Sigma$

wobei $A \subseteq \{0,1\}^*$, A abgeschlossen unter Präfixbildung und $w0 \in A \Leftrightarrow w1 \in A$

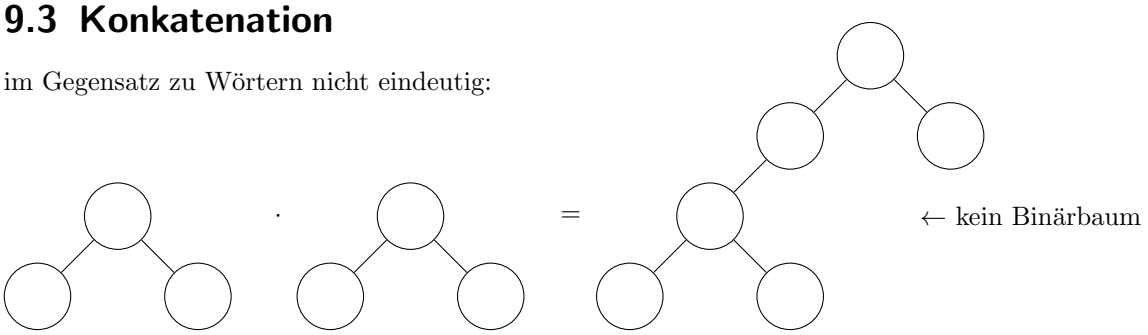
9.2.1 Beispiel



z.B. $t(01) = a$

9.3 Konkatenation

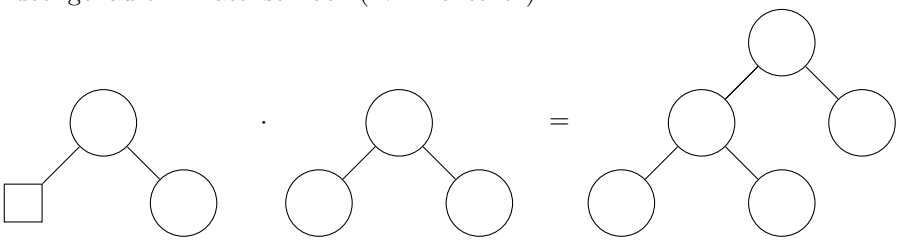
im Gegensatz zu Wörtern nicht eindeutig:



← kein Binärbaum

9.3.1 Kontexte

Idee: genau ein Blatt ist Loch ($\rightarrow$ "Kontext")

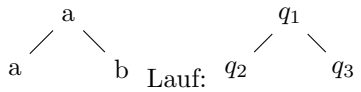


9.4 Baum-Automaten

- analog zu DFA/NFA
- bei Wörtern:

$$q_0 \overset{a}{\rightharpoonup} q_1 \overset{\dots}{\rightharpoonup} \dots \overset{\dots}{\rightharpoonup} q_n \quad \overset{?}{\in} F$$

- bei Bäumen: Lauf ist ein Baum



- es existieren bottom-up und top-down Automaten
- wir besprechen bottom-up

9.4.1 Definition

nicht-deterministischer bottom-up Baumautomat:
$\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$

- Q Zustände
- Σ Alphabet
- q_0 Startzustand
- $\delta \subseteq Q \times Q \times \Sigma \times Q$ Übergangsrelation
  - deterministisch: $\delta : Q \times Q \times \Sigma \to Q$
  - top-down: $\delta \subseteq Q \times \Sigma \times Q \times Q$
- $F \subseteq Q$ Endzustände

Sei $t \in T_\Sigma^*$. Ein Lauf ρ auf t erfüllt

$$(\rho(w0), \rho(w1), t(w), \rho(w)) \in \delta,\ t : A \to \Sigma,\ \rho : A \to Q$$

Semantik

$L(\mathcal{A}) = \{t \in T_\Sigma^* \mid \exists \rho \text{ Lauf auf } t : \rho(\epsilon) \in F \wedge w \text{ Blatt } \Rightarrow \rho(w) = q_0\}$

Determinisierung

Analog zu NFA→DFA: Potenzmengenkonstruktion
→ Komplementabschluss

Definition

Baumsprache heißt regulär, wenn sie von Baumautomaten akzeptiert wird.

9.5 Logik: MSO und FO auf Bäumen

9.5.1 Syntax

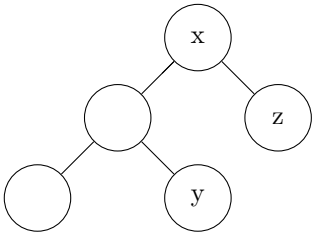
- $Q_a x$
- $x < y$
- $\neg\varphi$
- $\varphi \wedge \psi$
- $\exists x : \varphi$
- $\exists X : \varphi$

Domäne

- bei Wörtern: $\{0, \ldots, n\}$
- bei Bäumen: $A \subseteq \{0,1\}^*$ (wie oben)

Prädikat $x < y, +1$ sinnvoll

Beispiel



$x < z,\ x < y$ aber nicht $z < x$ oder $y < z$

9.5.2 Satz

Für Baumsprachen gilt $MSO = \text{Reg}$. Beweis analog zu Wörtern.

9.6 Baum-Grammatik

9.6.1 Definition

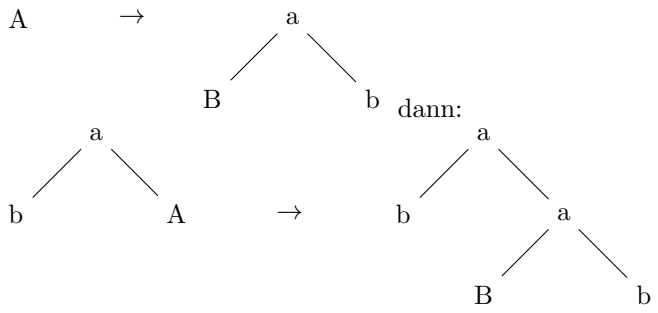
$G = (V, \Sigma, P, S)$

- V Variablen
- Σ Alphabet
- $S \in V$ Startsymbol

- $P = \{A \to t \mid A \in V,\ t \in T^*_\Sigma\}$ wobei V in t nur an Blättern erlaubt ist.

$L(G) = \{t \in T^*_\Sigma \mid S \overset{P^*}{\to} t\}$

Beispiel



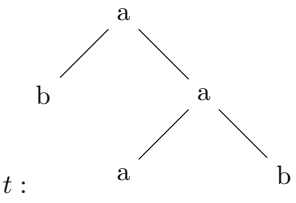
9.6.2 Satz

Baumsprache L regulär $\Leftrightarrow \exists$ Grammatik G mit $L(G) = L$

9.6.3 Definition: yield

Sei $t \in T^*_\Sigma$, dann ist $yield:\ T^*_\Sigma \to \Sigma^*$ die Blattsprache.

Beispiel



$yield(t) = bab$

9.6.4 Satz

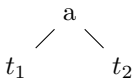
L kontextfrei $\Leftrightarrow \exists K \subseteq T^*_\Sigma$ reg mit $yield(K) = L$

Anmerkung

- Bäume, reguläre Sprachen: NC^1
- CFL, *yield*s: SAC^1

9.6.5 Beweis

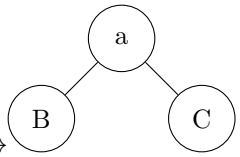
$\Leftarrow$: Sei G' Grammatik mit $L(G') = K$. Konstruiere kontextfreie Grammatik G mit $L(G) = L = yield(K)$. $G' = (V, \Sigma, S, P')$, dann $G = (V, \Sigma, S, P)$ mit $P = \{A \to yield(t) \mid A \to t \in P'\}$.
Es ist $yield(t) = yield(t_1)yield(t_2)$, wenn $t =$



Somit gilt für Satzform:
$t \overset{P'}{\to} t' \Leftrightarrow yield(t) \overset{P}{\to} yield(t')$
induktiv gilt: $t \in L(G') \Leftrightarrow yield(t) \in L(G)$

$\Rightarrow$ Sei L CFL.
$\Rightarrow \exists G = (V, \Sigma, P, S)$ in CNF mit $L = L(G)$
Konstruiere G' mit $yield(L(G')) = L$
Für jede Regel in G

- der Form $A \to a$ füge in G' ein $A \to$ 

- der Form $A \to BC$ füge in G' ein $A \to$ 

induktiv folgt $yield(L(G')) = L$ $\square$

10 Visibly-Counter-Sprachen & AC0

Quelle:
Visibly Counter Languages and Constant Depth Circuits, Krebs, Lange, Ludwig

10.1 Rückblick

Was wir von reg. Sprachen wissen:

- Reg ist NC^1-vollst.
- Reg $\cap\, AC^0 = FO[Reg.]$ = quasi aper. $\gamma : \Sigma^* \to Synt(L)$
- $L \in$ Reg $\setminus\, AC^0 \Rightarrow L$ ist ACC^0-schwer (Dichotomie)
- $AC^0 \subsetneq ACC^0_k \subseteq TC^0 \subseteq NC^1 \subseteq L \subseteq NL$

  ACC^0_k

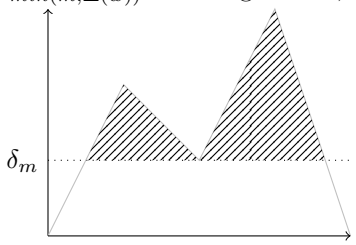
  $\emptyset \longleftarrow$ keine regulären Sprachen hier (Dichotomie)

  AC^0
- VPL auch NC^1-vollständig, also auch VCL.

10.2 Definition

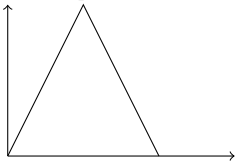
$m - VCA:\ \mathcal{A} = (Q, q_0, E, \Sigma, \delta_0, \ldots, \delta_m)$ mit $\delta_i = Q \times \Sigma \to \Sigma$
(i je nach Ebene, für Ebene $\geq m$ immer δ_m)

Akzeptanzkriterium: Endzustand in Lauf erreichbar
$\delta_{min(m,\Delta(w))}$ wird angewandt, nachdem w gelesen wurde. $\Delta(w) = |w|_{call} - |w|_{return}$

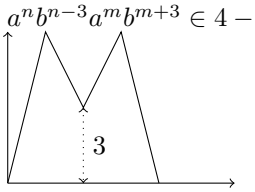


10.2.1 Beispiel

- $a^n b^n \in 1 - VCL$



- $a^n b^{n-3} a^m b^{m+3} \in 4 - VCL$



10.2.2 Anmerkung

Nicht in AC^0:

- $\mathbb{D}$, da TC^0-schwer
- MAJ, EQU (Anzahl a, b vergleichen)

Beispiel

$L = (a|aba)^n b^n$ ist TC^0-schwer

Beweis:

zeige $EQU \leq_{AC^0} L$

$f \in AC^0$ mit $w \in EQU \Leftrightarrow f(w) \in L$

$f(w) = \varphi(w)\psi(w) \quad \varphi, \psi$ Homs.

$\varphi(a) = aaa$

$\varphi(b) = aba \quad \psi(w) = b^{2|w|}$

Zu viele a's:  zu viele b's: 

$$|w|_a = |w|_b \to |f(w)|_a = \frac{3|w|}{2} + \frac{2|w|}{2} = \frac{5}{2}|w|$$

$$|f(w)|_b = \frac{|w|}{2} + 2|w| = \frac{5}{2}|w|$$

10.3 Was macht L schwer?

→ Variable Steigung
Bsp. kann verallgemeinert werden

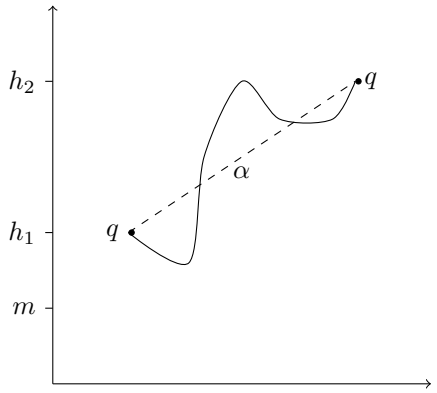
10.3.1 Definition

Zustand q hat konstante Steigung, wenn $\exists \alpha \in \mathbb{Q}$ (α: Steigung) $\forall w \in \Sigma^*$ mit

$$(q, h_1) \xrightarrow{w} (q, h_2) \text{ und}$$

$\forall w'$ Präfix von $w : h_1 + \Delta(w') \geq m$ gilt:

$$h_2 = \alpha|w| + h_1 \text{ Steigung}$$



Äquivalent ("Korridor"):

$$\exists \alpha \in \mathbb{Q}, \gamma \in \mathbb{N} \forall w \in \Sigma^*, w' \text{ Präfix von } w : \alpha|w'| - \gamma \leq \Delta(w') \leq \alpha|w'| + \gamma$$

10.4 Einfaches Höhenverhalten

10.4.1 Definition: aktiver Zustand

Zustand q heißt *aktiv*, wenn $\exists w \in L(\mathcal{A})$ und Position $i < j$, sodass $(q_0, 0) \xrightarrow{w_1 \ldots w_i} (q, h), h > m + |Q|$ und $h - \Delta(w_1 \ldots w_j) > |Q|$

10.4.2 Lemma

Für $L = L(\mathcal{A})$, sodass $\mathcal{A}$ aktiven Zustand ohne konstante Steigung hat, dann $L \notin AC^0$ (TC^0-schwer)
Beweis: verallgemeinertes Beispiel von oben.

10.4.3 Definition

Hat ein VCA $\mathcal{A}$ konstante Steigung in allen aktiven Zuständen, so hat die Sprache $L(\mathcal{A})$ *einfaches Höhenverhalten (EHV)*

10.4.4 Definition: Höhentransduktion

$\mathcal{T}_m(w) : \Sigma^* \to \Sigma_m^*, \Sigma_m = \Sigma \times \{0, \ldots, m\}$, $\Delta_m(w) = \min(m, \Delta(w))$
$\mathcal{T}_m(w) = (w_1, \Delta_m(\epsilon))(w_2, \Delta_m(w_1))(w_3, \Delta_m(w_1 w_2)) \ldots (w_n, \Delta(w_1 \ldots w_{n-1}))$
Beispiel: $\mathcal{T}_2(aaba) = (a,0)(a,1)(b,2)(a,1)$
$w \in \Sigma_m^*$ heißt gültig, wenn $w \in F(\mathcal{T}_m(\Sigma^*))$ (F Faktor)

10.4.5 Definition

gegeben m-VCA $\mathcal{A} = (Q, q_0, E, \Sigma, \delta_0, \ldots, \delta_m)$ dann $R_\mathcal{A} = L(M)$ mit endlichem Automat $M = (Q, q_0, E, \Sigma_m, \delta)$ mit $\delta(q, (a,i)) = \delta_i(q, a)$

10.4.6 Lemma

$\mathcal{A}$ ist m-VCA, dann $w \in L(\mathcal{A}) \Leftrightarrow \mathcal{T}_m(w) \in R_\mathcal{A}$
Also: Wortproblem von $L(\mathcal{A})$ zweiteilbar:

- berechne $\mathcal{T}_m(w)$
- teste ob in $R_\mathcal{A}$

Bisher: $\neg EHV \Rightarrow \notin AC^0$ (da $\mathcal{T}_m \notin AC^0$)
Jetzt: $EHV \Rightarrow \mathcal{T}_m \in AC^0$ berechenbar

10.4.7 Satz

EHV$\Rightarrow$ Höhenprädikat $H_m(x)$ in $FO[+]$ definierbar

- $w_{x=i} \models H_k(x) \Rightarrow \Delta(w_1 \ldots w_{i-1}) = k$ falls $w \in \Sigma^*$
- $w_{x=i} \models H_k(x) \Leftrightarrow \Delta(w_1 \ldots w_{i-1}) = k$ falls $w \in L$

Beweisidee

EHV $\xrightarrow{1.}$ Matching-Prädikat $\xrightarrow{2.}$ Höhenprädikat

1. $M(x,y)$ ist wahr, wenn well-matched zwischen x und y
   Betrachte Fall große Höhe und aktive Zustände:

   - Fasse Schleifen zusammen:
     Bsp.: $q_1 \underbrace{q_2 q_3 q_2 q_4 q_2}_{q_2} q_5 q_6 \rightsquigarrow q_1 q_2 q_5 q_6$ (jeder Zustand höchstens $1\times \Rightarrow$ endliche Kombinationsmöglichkeiten $|Q|!$ für komprimierte Läufe)

- Jeder Zustand deckt Bereich mit konstanter Steigung ab ⇒ Wort ist aufteilbar in höchstens $|Q|!$ Abschnitte konstanter Steigung
- Formel für $M(x,y)$: rate Lauf und verifiziere Steigungen ($n = |Q|$):
  $M(x,y) = x < y \wedge \Sigma_{push}(x) \wedge \Sigma_{pop}(y) \wedge$
  $\bigvee_{(q_1,\ldots,q_n)\in Q^n} \Big(\exists z_0,\ldots,z_n \exists h_0,\ldots,h_n : z_0 = x \wedge z_n = y-1 \wedge h_0 = h_n \wedge$
  $\bigwedge_{i=0}^{n-1} z_i \leq z_{i+1} \wedge A_{q_{i+1}}(z_i, z_{i+1}, h_i, h_{i+1}, h_0) \Big)$ (Definition von $A_{q_{i+1}}$: Paper)

10.5 Der reguläre Anteil einer VCL

$\exists \mathcal{A} : R_\mathcal{A} \in AC^0 \Rightarrow \exists FO[reg]$-Formel
$\forall \mathcal{A} : R_\mathcal{A} \notin AC^0 \Rightarrow L(\mathcal{A}) \notin AC^0$

10.5.1 Beispiel

$L = \{wb^{|w|} \mid w \in \{a_1, a_2\}, |w|_{a_1} \equiv 0 \pmod 2\}$
$\mathcal{A}$ mit $L(\mathcal{A}) = L$, dann "$R_\mathcal{A} = Parity(a_1, a_2)b^*$"
$L' = L \cap \Sigma^{10} \in AC^0$ ($L(R_{\mathcal{A}'}) = L'$)
aber $R_{\mathcal{A}'}$ könnte:

- endlich sein ⇒ $R_{\mathcal{A}'} \in AC^0$
- "$R_{\mathcal{A}'} = Parity(a_1, a_2)b^{10}$"⇒ $R_{\mathcal{A}'} \notin AC^0$

⇒ Lösung: Normalform
Problem: Schleife, die $R_\mathcal{A}$ komplex macht aber unnötig ist.

10.5.2 Definition (informell)

$\mathcal{A}$ heißt Schleifen-normal, wenn jede Schleife synchron pumpbar ist.

10.5.3 Lemma

Für jeden VCA $\mathcal{A}$ existiert Schleifen-normaler Automat $\mathcal{A}'$ mit $L(\mathcal{A}) = L(\mathcal{A}')$

10.5.4 Lemma

Ist $\mathcal{A}$ Schleifen-normaler m-VCA, dann:
$\exists t > 0\ \exists G \subseteq \Sigma_m^t$ mit $G^* \subseteq F(\mathcal{T}_m(\Sigma^*))$, sodass $\eta_{R_\mathcal{A}}(G)$ nicht-triviale Gruppe enthält, dann $L(\mathcal{A}) \notin AC^0$ (vgl. quasi-aperiodisch)

10.5.5 Satz (Wdh.)

Ist L regulär, dann $L \in AC^0 \Leftrightarrow \forall t > 0 \forall G \subseteq \Sigma^t : \eta_L(G)$ hat nur triviale Gruppen $\Leftrightarrow L \in FO[reg]$

10.5.6 Lemma

$\forall t > 0 \forall G \subseteq \Sigma_m^t$ mit $G^* \subseteq F(\mathcal{T}_m(\Sigma^*))$, so dass $\eta_{R_\mathcal{A}}$ hat nur triviale Teilgruppen $\Rightarrow \exists FO[reg]$-Formel φ mit $L(\varphi) \cap \mathcal{T}_m(\Sigma^*) = R_\mathcal{A} \cap \mathcal{T}_m(\Sigma^*)$

10.5.7 Satz

Sei $\mathcal{A}$ ein Schleifen-normaler m-VCA für $L = L(\mathcal{A})$.
$L \in AC^0 \Leftrightarrow L$ hat EHV und $\forall t > 0 \forall G \subseteq \Sigma_m^t$ mit $G^* \subseteq F(\mathcal{T}_m(\Sigma^*))$ hat $\eta_{R_\mathcal{A}}$ nur triviale Teilgruppen.

Beweis

Beginne mit $FO[reg]$-Formel φ mit $\underbrace{L(\varphi)}_{\subseteq \Sigma_m^*} \cap \mathcal{T}_m(\Sigma^*) = R_\mathcal{A} \cap \mathcal{T}_m(\Sigma^*)$.

$Q_{(a,i)}x$-Prädikate in φ ersetzen durch

- $(Q_a x \wedge H_i x)$, falls $i < m$
- $(Q_a x \wedge H_{\geq m} x)$, falls $i \geq m$

□

Alternative Sichtweise: $\mathcal{T}_m^{-1}(L(\varphi)) = \mathcal{T}_m^{-1}(R_\mathcal{A})$

10.5.8 Satz

Gegeben $\mathcal{A}$, dann ist es entscheidbar, ob $L(\mathcal{A}) \in AC^0$

Beweis

Betrachte Schleifen-normalen Automaten $\mathcal{A}'$

- Überprüfe EHV: $\forall q \in Q$: "q aktiv $\Rightarrow$ konstante Steigung"
  - konstante Steigung: teste $w \in \Sigma^{\leq |Q|}$, die durch q schleifen. Haben alle die gleiche Steigung?
  - aktiv: finde x mit $\Delta(x) > m + |Q|$ und y mit $xy \in L(\mathcal{A})$ mit $y = y'y''$ mit $\Delta(y') > -|Q|$
    - "x finden" auf Leerheitsproblem für VCL reduzierbar
    - $y : \Delta(x) \cdot |Q|$ maximale Länge
- regulärer Anteil: analog quasi-aperiodisch für reguläre Sprachen. □

10.6 Anmekung

$VPL \cap AC^0$ noch offen

10.6.1 Beispiel

- $S \to a_1Sb|a_2Sbc|\epsilon$
- $S \to aSbc|acSb|\epsilon$
  unbekannt ob $\in AC^0$

vgl. $S \to aSb|aSbc|\epsilon$ ($\in VCL,\ \notin AC^0$)