Theoretische Informatik

Nicola Studer
nicstuder@student.ethz.ch

April 5, 2023

1 Gruppentheorie

Def 1.1 (Monoid)**.** $\langle M; *, e\rangle$ mit $*$ assoziativ und e als neutrales Element.

Def 1.2 (Gruppe)**.** $\langle G; *,' e\rangle$ mit $*$ assoziativ, e als neutrales Element und jedes element $x \in G$ hat Inverses $\hat{x}$.

2 Alphabete, Wörter, Sprachen und die Darstellung von Problemen

Def 2.1. Eine endliche nichtleere Menge Σ heisst **Alphabet**. Die Elemente eines Alphabets heissen **Buchstaben** (**Zeichen**, **Symbole**).

Bsp. $\Sigma_{\text{bool}} = \{0, 1\}$, $\Sigma_{\text{lat}} = \{a, b, c, \ldots, z\}$,
$\Sigma_{\text{Keyboard}} = \Sigma_{\text{lat}} \cup \{A, B, \ldots, Z, \textvisiblespace, >, <, (,), \ldots, !\}$,
$\Sigma_{\text{logic}} = \{0, 1, x, (,), \wedge, \vee, \neg\}$

Def 2.2. Ein **Wort** über Σ ist eine endliche (eventuell leere) Folge von Buchstaben aus Σ. Das **leere Wort** λ ist die leere Buchstabenfolge. $|\lambda| = 0$.
$\boldsymbol{\Sigma^*}$ ist die Menge aller Wörter über Σ, $\Sigma^+ = \Sigma^* - \{\lambda\}$.
$\boldsymbol{\Sigma^i} = \{x \in \Sigma^* \mid |x| = i\}$

Def 2.3 (Verkettung)**.** $\text{Kon} : \Sigma^* \times \Sigma^* \to \Sigma^*$.

$$\text{Kon}(x, y) = x \cdot y = xy$$

Bmk 2.1. (Σ^*, Kon) ist Monoid mit neutralem Element λ.

Bmk 2.2. $\forall x, y \in \Sigma^* : |xy| = |x \cdot y| = |x| + |y|$.

Def 2.4 (Umkehrung)**.** $a^{\text{R}} = a_n a_{n-1} \ldots a_1$

Def 2.5. i-te Iteration x^i von $x \in \Sigma$ wird definiert als $x^0 = \lambda$, $x^1 = x$ und $x^i = xx^{i-1}$.

Def 2.6.

- v heisst **Teilwort** von w $\iff$ $\exists x, y \in \Sigma^* : w = xvy$.
- v heisst **Präfix** von w $\iff$ $\exists y \in \Sigma^* : w = vy$.
- v heisst **Suffix** von w $\iff$ $\exists x \in \Sigma^* : w = xv$.
- $v \neq \lambda$ heisst **echtes Teilwort** (Präfix, Suffix) von w genau dann, wenn $v \neq w$ und v ein Teilwort (Präfix, Suffix) von w ist.

Def 2.7. $|x|_a$ Anzahl der Vorkommen von a in x.

Def 2.8. $\Sigma = \{s_1, s_2, \ldots, s_m\}$, $m \geq 1$ mit $s_1 < s_2 \ldots < s_m$ als Ordnung auf Σ. Wir definieren die **kanonische Ordnung** auf Σ^* für $u, v \in \Sigma^*$:

$$u < v \iff |u| < |v| \vee |u| = |v| \wedge u = x \cdot s_i \cdot u' \wedge v = x \cdot s_j \cdot v'$$

für irgendwelche $x, u', v' \in \Sigma^*$ und $s_i < s_j$.

Def 2.9 (Sprache)**.** $L \subseteq \Sigma^*$ mit Komplement $L^{\text{C}} = \Sigma^* - L$.

- $\boldsymbol{L_\emptyset} = \emptyset$ ist die **leere Sprache**
- $\boldsymbol{L_1 \cdot L_2 = L_1 L_2} = \{vw \mid v \in L_1 \text{ und } w \in L_2\}$
- $\boldsymbol{L^0} := L_\lambda$ und $\boldsymbol{L^{i+1}} = L^i \cdot L$ für alle $i \in \mathbb{N}$
- $\boldsymbol{L^*} = \bigcup_{i \in \mathbb{N}} L^i$ und $\boldsymbol{L^+} = \bigcup_{i \in \mathbb{N} - \{0\}} L^i = L \cdot L^*$

Lemma 2.1. $L_1 L_2 \cup L_1 L_3 = L_1(L_2 \cup L_3)$

Lemma 2.2. $L_1(L_2 \cap L_3) \subseteq L_1 L_2 \cap L_1 L_3$

Lemma 2.3. Es existieren $U_1, U_2, U_3 \in (\Sigma_{\text{bool}})^*$, so dass $U_1(U_2 \cap U_3) \subsetneq U_1 U_2 \cap U_1 U_3$

Def 2.10. Σ_1, Σ_2 zwei beliebige Alphabete. Homomorphismus: $h : \Sigma_1^* \to \Sigma_2^*$ mit:

1. $h(\lambda) = \lambda$
2. $h(uv) = h(u) \cdot h(v)$ für alle $u, v \in \Sigma_1^*$.

Def 2.11. Das **Entscheidungsproblem** $(\boldsymbol{\Sigma, L})$ ist, für jedes $x \in \Sigma^*$ zu entscheiden, ob $x \in L$ oder $x \notin L$.

Ein Algorithmus A löst das Entscheidungsproblem (**erkennt** L), falls $\forall x \in \Sigma^*$:

$$A(x) = \begin{cases} 1, & \text{falls } x \in L \\ 0, & \text{falls } x \notin L \end{cases}$$

Def 2.12. Σ, Γ zwei Alphabete. Algorithmus A **berechnet (realisiert)** eine **Funktion (Transformation)** $\boldsymbol{f : \Sigma^* \to \Gamma^*}$ falls $\forall x \in \Sigma^* : A(x) = f(x)$.

Def 2.13. Σ, Γ zwei Alphabete, $R \subseteq \Sigma^* \times \Gamma^*$ eine Relation. Ein Algorithmus $\boldsymbol{A}$ **berechnet** $\boldsymbol{R}$ (oder löst das Relationsproblem R), falls für alle $x \in \Sigma^*$, für das ein $y \in \Gamma^*$ nut $(x, y) \in R$ exisitert gilt: $(x, A(x)) \in R$.

Def 2.14 (Optimierungsproblem)**.** Ist ein 6-Tupel $\mathcal{U} = (\Sigma_1, \Sigma_0, L, \mathcal{M}, \text{cost}, \text{goal})$, wobei:

1. Σ_1 ist das Eingabealphabet
2. Σ_2 ist das Ausgabealphabet
3. $L \subseteq \Sigma^*$ ist die Sprache der **zulässigen Eingaben**. $x \in L$ ist ein **Problemfall (Instanz) von** $\boldsymbol{\mathcal{U}}$.
4. $\mathcal{M} : L \to \mathcal{P}(\Sigma_0^*)$ und für jedes $x \in L$ ist $\mathcal{M}(x)$ die **Menge der zulässigen Lösungen für** $\boldsymbol{x}$.
5. **Kostenfunktion** $\text{cost} : \bigcup_{x \in L}(\mathcal{M}(x) \times \{x\}) \to \mathbb{R}^+$
6. **Optimierungsziel** *goal* $\in \{Minimum, Maximum\}$

Def. Eine zulässige Lösung $\alpha \in \mathcal{M}(x)$ heisst **optimal** für den Problemfall x, falls

$$\text{cost}(\alpha, x) = \textbf{Opt}_{\mathcal{U}}(x) = \text{goal}\{\text{cost}(\beta, x) \mid \beta \in \mathcal{M}(x)\}$$

Def. Ein Algorithmus A **löst** $\mathcal{U}$, falls für jedes $x \in L$

1. $A(x) \in \mathcal{M}(x)$
2. $\text{cost}(A(x), x) = \text{goal}\{\text{cost}(\beta, x) \mid \beta \in \mathcal{M}(x)\}$

Def 2.15. Algorithmus A **generiert** das Wort x, falls A für die Eingabe λ die Ausgabe x liefert.

Def 2.16. A ist ein **Auszählungsalgorithmus für** L, falls A für jede Eingabe $n \in \mathbb{N}$ die Wortfolge $x_1, x_2, \ldots, x_n$ ausgibt, wobei das die kanonisch n erster Wörter in L sind.

Shannon Entropie

Claude Shannon versuche den Informationsgehalt von String herauszufinden, hat es aber nicht geschafft Positionen darzustellen.

Betrachte das Alphabet $\Sigma = \{a, b, c, d\}$ und das Wort mit Vorkommen $a \times 16, b \times 8, c \times 4, d \times 4$. Seine Codierung ist $a \mapsto 0, b \mapsto 10, c \mapsto 110, d \mapsto 111$. Man erkennt das nur die Präfixe erweitert werden.

Die Codierung hat nun die Folgende Länge mit $|w|$ als Wortlänge, H_a Häufigkeit von a, h_a relative Häufigkeit und p_a die Wahrscheinlichkeit:

$$\begin{aligned}
56 &= 16 \cdot 1 + 8 \cdot 2 + 4 \cdot 3 + 4 \cdot 3 \\
&= H_a \log_2 \frac{|w|}{H_a} + H_b \log_2 \frac{|w|}{H_b} + \dots \\
&= H_a \cdot (-\log_2 \frac{H_a}{|w|}) + \dots \\
&= H_a \cdot (-\log_2 h_a) + \dots \\
&= H_a \cdot (-\log_2 p_a) + \dots \\
&= -\sum_{a \in \Sigma} H_a \cdot \log_2 p_a
\end{aligned}$$

Kolmogorov-Komplexität

Def 2.17. Für jedes Wort $x \in \{0,1\}^*$ ist die $K(x)$ des Wortes das Minimum der binären Länge der Pascal-Programme, die x generieren.

Lemma 2.4. $\exists d \in \mathbb{N}\ \forall x \in \{0,1\}^* : K(x) \leq |x| + d$

Def 2.18. $\forall n \in \mathbb{N} : K(n) = K(\text{Bin}(n))$

Lemma 2.5. $\forall n \in \mathbb{N}_1\ \exists w_n \in \{0,1\}^n : K(w_n) \geq |w| = n$

Thm 2.1. Sei A und B Programmiersprachen. Es existiert eine Konstante $c_{A,B}$ die nur von A und B abhängt, so dass $\forall x \in \{0,1\}^* : |K_A(x) - K_B(x)| \leq c_{A,B}$

Def 2.19. $x \in \{0,1\}^*$ heisst zufällig $\iff K(x) \geq |x|$.

Eine Zahl n heisst zufällig $\iff K(n) = K(\text{Bin}(n)) \geq \lceil \log_2(n+1) \rceil - 1$. Die -1 kommt von der führenden 1, welche bei jeder Binären Zahl führend ist.

Thm 2.2. $L \subseteq \{0,1\}^*$, sei z_n das kanonisch n-te Wort in L. Wenn ein Programm A_L existiert, das das $\{0,1\}^*, L$ löst, dann gilt für alle $n \in \mathbb{N} - \{0\}$: $K(z_n) \leq \lceil \log_2(n+1) \rceil + c$, wobei c eine von n unabhängige Konstante ist.

Thm 2.3 (Primzahlsatz)**.**

$$\lim_{n \to \infty} \frac{\text{Prim}(n)}{n / \ln n} = 1$$

Lemma 2.6. Sei $n_1, n_2, n_3, \dots$ eine steigende unendliche folge natürlicher Zahlen mit $K(n_i) \geq \lceil \log_2 n_i \rceil / 2$. Für jedes $i \in \mathbb{N} - \{0\}$ sei q_i die grösste Primzahl, die die Zahl n_i teilt. Dann ist die Menge $Q = \{q_i \mid i \in \mathbb{N} - \{0\}\}$ unendlich.

Thm 2.4. Für unendlich viele $k \in \mathbb{N}$ gilt

$$\text{Prim(k)} \geq \frac{k}{2^{17} \log_2 k \cdot (\log_2 \log_2 k)^2}$$

3 Endliche Automaten

Def 3.1 (*deterministischer* endlicher Automat)**.**
$M = (q, \Sigma, \delta, q_0, F)$ mit:

1. Q: endliche Menge von Zuständen
2. Σ: Eingabealphabet
3. $q_0 \in Q$: der Anfangszustand
4. $F \subseteq Q$: Menge der akzeptierenden Zustände
5. $\delta : Q \times \Sigma \to Q$: Übergangsfunktion

Def (Weitere Terminologie)**.**

1. **Konfiguration** von M ist $x \in Q \times \Sigma^*$.
2. **Startkonfiguration** von M auf x ist $(q_0, x) \in \{q_0\}$
3. **Endkonfiguration**: Jede Konfiguration von $Q \times \{\lambda\}$
4. **Schritt** von M: Relation $\vdash_M \subseteq (Q \times \Sigma^*) \times (Q \times \Sigma^*)$ definiert durch: $(q, w) \vdash_M (p, x) \iff w = ax, a \in \Sigma$ und $\delta(q, a) = p$
5. **Berechnung** C von M: Folge Folge von Konfigurationen $C = C_0, \dots, C_n$ so dass $\forall 0 \leq i < n : C_i \vdash_M C_{i+1}$
6. **Berechnung von M auf Eingabe $x \in \Sigma^*$** : Falls $C_0 = (q_0, x)$ und $C_n \in Q \times \{\lambda\}$
7. **Akzeptierende Berechnung von** M **auf** x: Falls $C_n \in F \times \{\lambda\}$ und M das Wort x akzeptiert.
8. **Verwerfende Berechnung von** M **auf** x: Falls $C_n \in (Q - F) \times \{\lambda\}$ und M das Wort x verwirft.
9. **Die von** M **akzeptierte Sprache** $L(M) =$

$$\{w \in \Sigma^* \mid \text{Die Berechnung von } M \text{ auf } w \text{ endet in einer Endkonfiguration } (q, \lambda) \text{ mit } q \in F$$

10. **Klasse der regulären (akzeptierten) Sprachen**: $\mathcal{L}_{\text{EA}} = \{L(M) \mid M \text{ ist ein EA}\}$

Def 3.2. $(q, w) \vdash_M^* (p, u) \iff (q = p \land w = u) \lor k \in \mathbb{N}_1$:

(i) $w = a_1 a_2 \dots a_k u, a_i \in \Sigma$ für $i = 1, 2, \dots, k$

(ii) $\exists r_1, r_2, \dots, r_{k-1} \in Q$, so dass $(q, w) \vdash_M (r_1, a_2 \dots a_k u) \dots \vdash_M (r_{k-1}, a_k u) \vdash_M (p, u)$

Def. $\hat{\delta} : Q \times \Sigma^* \to Q$ durch:

(i) $\forall q \in Q : \hat{\delta}(q, \lambda) = q$

(ii) $\forall a \in \Sigma, w \in \Sigma^*, q \in Q : \hat{\delta}(q, wq) = \delta(\hat{\delta}(q, w), a)$

Lemma 3.1. $L(M) = \{w \in \{0,1\}^* \mid |w|_0 + |w|_1 \equiv_2 0$

Def. $\text{Kl}[p] =$
$\{w \in \Sigma^* \mid \hat{\delta}(q_0, w) = p\} = \{w \in \Sigma^* \mid (q_0, w) \vdash_M^* (p, \lambda)\}$

Lemma 3.2. $M_1 = (Q_1, \Sigma, \delta_1, q_{01}, F_1)$, sowie auch $M_2 = (Q_2, \Sigma, \delta_2, q_{02}, F_2)$ sind zwei EA, es gilt: $\forall \odot \in \{\cup, \cap, -\}\ \exists M : L(M) = L(M_1) \odot L(M_2)$

Konstruktion von M: $M = (Q, \Sigma, \delta, q_0, F_\odot)$

(i) $Q = Q_1 \times Q_2$

(ii) $q_0 = (q_{01}, q_{02})$

(iii) $\forall q \in Q_1 p \in Q_2\ a \in \Sigma : \delta((q, p), a) = (\delta_1(q, a), \delta_2(p, a))$

(iv) $\odot = \cup \implies F = F_1 \times Q_2 \cup Q_1 \times F_2$
$\odot = \cap \implies F = F_1 \times F_2$
$\odot = - \implies F = F_1 \times (Q_2 - F_2)$

Lemma 3.3. $\forall A = (Q, \Sigma, \delta_A, q_0, F)\ \forall x, y \in \Sigma^*, x \neq y$:

$$\begin{aligned}
&(q_0, x) \vdash_A^* (p, \lambda) \land (q_0, y) \vdash_A^* (p, \lambda) \\
&\implies \forall z \in \Sigma^*\ \exists r \in Q : xz \in \text{Kl}[r] \land yz \in \text{Kl}[r] \\
&\implies xy \in L(A) \iff yz \in L(A)
\end{aligned}$$

Lemma 3.4 (Pumping-Lemma)**.**
$\forall L \in \mathcal{L}_{EA}\ \exists n_0 \in \mathbb{N}\ \forall w \in \Sigma^*, |w| \geq n_0\ \exists yxz = wl$:

1. $|yx| \leq n_0$
2. $|x| \geq 1$
3. $\{yx^k \mid k \in \mathbb{N}\} \lor \{yx^k z \mid k \in \mathbb{N}\} \cup L = \emptyset$

Thm 3.1. $\forall L \subset \{0,1\}^* \in \mathcal{L}_{EA}\ \forall x \in \{0,1\}^*\ \exists c : \forall n$-te Wort y in $L_x : K(y) \leq \lceil \log_2(n+1) \rceil + c$ mit $L_x = \{y \in \Sigma^* \mid xy \in L\}$

Cor. $\forall L \subset \{0,1\}^* \in \mathcal{L}_{EA}\ \forall x \in \{0,1\}^*\ \exists c : \forall$*erste Wörter* y *in* L_x*:* $K(y) \leq c$

Nichtdeterminismus

Def 3.3 (*nicht deterministischer* endlicher Automat)**.**
$M = (q, \Sigma, \delta, q_0, F)$ mit:

1. Q: endliche Menge von Zuständen
2. Σ: Eingabealphabet
3. $q_0 \in Q$: der Anfangszustand
4. $F \subseteq Q$: Menge der akzeptierenden Zustände
5. $\delta : Q \times \Sigma \to \mathcal{P}(Q)$: Übergangsfunktion

Def (Weitere Terminologie)**.**

1. **Konfiguration** von M ist $x \in Q \times \Sigma^*$.
2. **Startkonfiguration** von M auf x ist $(q_0, x) \in \{q_0\}$
3. **Endkonfiguration**: Jede Konfiguration von $Q \times \{\lambda\}$
4. **Schritt** von M: Relation $\vdash_M \subseteq (Q \times \Sigma^*) \times (Q \times \Sigma^*)$ definiert durch:
   $(q, w) \vdash_M (p, x) \iff w = ax, a \in \Sigma$ und $p \in \delta(q, a)$
5. **Berechnung** C von M: Folge Folge von Konfigurationen $C = C_0, \ldots, C_n$ so dass $\forall 0 \leq i < n : C_i \vdash_M C_{i+1}$
6. **Berechnung von M auf Eingabe $x \in \Sigma^*$** : Falls $C_0 = (q_0, x)$ und $C_n \in Q \times \{\lambda\}$
7. **Akzeptierende Berechnung von** M **auf** x: Falls $C_n = (p, \lambda), p \in F$ und M das Wort x akzeptiert.
8. **Verwerfende Berechnung von** M **auf** x: Falls $C_n \in (Q - F) \times \{\lambda\}$ und M das Wort x verwirft.
9. **Die von** M **akzeptierte Sprache** $L(M) = \{w \in \Sigma^* \mid (q_0, w) \vdash_M^* (p, \lambda)$ für ein $p \in F\}$
10. $\hat{\delta} : Q \times \Sigma^* \to \mathcal{P}(q)$
    (a) $\hat{\delta}(q, \lambda) = \{q\}$
    (b) $\hat{\delta}(q, wa) = \{p \mid \exists r \in \hat{\delta}(q, w) : p \in \delta(r, a)\}$
    (c) $\hat{\delta}(q, wa) = \bigcup_{r \in \hat{\delta}(q,w)} \delta(r, a)$ für alle $q \in Q, a \in \Sigma, w \in \Sigma^*$

Thm 3.2. Zu jedem *NEA* M existiert ein *EA* A, so dass $L(M) = L(A)$

1. $Q_A = \{\langle P \rangle \mid P \subseteq Q\}$
2. $\Sigma_A = \Sigma$
3. $q_{0A} = \langle \{q_0\} \rangle$
4. $F_A = \{\langle P \rangle P \mid P \subseteq Q \ \wedge \ P \cap F \neq \emptyset\}$
5. - $\delta_A = Q_A \times \Sigma \times Q_A$
   - $\delta_A(\langle P \rangle, a) = \left\langle \bigcup_{p \in P} \delta_M(p, q) \right\rangle = \langle \{p \in Q \mid \exists p \in P \ \wedge \ q \in \delta_M(p, a)\} \rangle$

Zahlen nachkorrigieren

Lemma 3.5. $L_k = \{x1y \mid x \in \{0,1\}^*, y \in \{0,1\}^{k-1}\}$
Für alle $k \in \mathbb{N}_1$ muss jeder EA, der L_k akzeptiert, mindestens 2^k Zustände haben.