Definition 2.4

Es sei Σ ein Alphabet, $x \in \Sigma^*$, $i \in \mathbf{N}$, $i > 0$.

Wir definieren:

$x^0 := \lambda, \quad x^i := xx^{i-1}$.

Beispiel

$(aba)^3 = abaabaaba$

Definition 2.5

Es seien $v, w \in \Sigma^*$.

1. v **ist Teilwort von** w, wenn es $x, y \in \Sigma^*$ gibt mit $w = xvy$.
2. v **ist Suffix von** w, wenn es $x \in \Sigma^*$ gibt mit $w = xv$.
3. v **ist Präfix von** w, wenn es $yx \in \Sigma^*$ gibt mit $w = vy$.
4. v ist **echtes** Teilwort (Präfix, Suffix) von w, wenn v Teilwort (Präfix, Suffix) von w ist und sowohl $v \neq \lambda$ als auch $v \neq w$ gilt.

Definition 2.6

Es sei $x \in \Sigma^*$ und $a \in \Sigma$. Dann ist $|x|_a$ die Anzahl der Vorkommen von a in x.

Beispiel

$|abaabba|_a = 4$

$|abaabba|_b = 3$

Definition 2.7

Es sei Σ ein Alphabet und $<$ eine Ordnung auf Σ.

Die **kanonische Ordnung** $<$ auf Σ^* ist so definiert:

$u < w$ gilt genau dann wenn

- $|u| < |v|$ oder
- $|u| = |v|$ und u ist lexikographisch kleiner als v

Definition 2.8

Es sei Σ ein Alphabet.

1. Ein Teilmenge $L \subseteq \Sigma^*$ heißt **Sprache (über Σ)**.
2. Das Komplement L^C oder $\overline{L}$ von L bezüglich Σ ist $\Sigma^* - L$.
3. $\emptyset$ ist die **leere Sprache**.
4. Sind L_1 und L_2 Sprachen über Σ, so ist $L_1 \cdot L_2 = L_1 L_2 = \{ vw \mid v \in L_1,\ w \in L_2 \}$.
5. Ist $L \subseteq \Sigma^*$, dann $L^0 := \{\lambda\}$, $L^{i+1} := L L^i$ für $i \in \mathbf{N}$.
6. $L^* = \bigcup_{i \geq 0} L^i$, $L^+ = \bigcup_{i > 0} L^i$.

Definition 2.9

Es seien Σ_1, Σ_2 Alphabete.

Ein **Homomorphismus** von Σ_1^* nach Σ_2^* ist eine Abbildung $h\colon \Sigma_1^* \to \Sigma_2^*$ mit

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

Definition 2.10

Das **Entscheidungsproblem** (Σ, L) für ein gegebenes Alphabet Σ und eine Sprache $L \subseteq \Sigma^*$ ist, für jedes $x \in \Sigma^*$ zu entscheiden, ob $x \in L$ oder $x \notin L$.

Ein Programm A **löst** das Entscheidungsproblem (Σ, L), falls für alle $x \in \Sigma^*$ gilt:

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

Wir sagen auch, daß A die Sprache A **erkennt**.

Definition 2.11

Seien Σ und Γ zwei Alphabete. Wir sagen, daß ein Algorithmus A eine **Funktion (Transformation)** $f\colon \Sigma^* \to \Gamma^*$ **berechnet (realisiert)**, falls $A(x) = f(x)$ für alle $x \in \Sigma^*$.

Definition 2.12

Seien Σ und Γ zwei Alphabete und sei $R \subseteq \Sigma^* \times \Gamma^*$ eine Relation in Σ^* und Γ^*. Ein Algorithmus A **berechnet** R **(löst das Relationsproblem** R**)**, falls für jedes $x \in \Sigma^*$ gilt, daß $(x, A(x)) \in R$.

Definition 2.13

Ein **Optimierungsproblem** ist ein 6-Tupel $\mathcal{U} = (\Sigma_I, \Sigma_O, L, \mathcal{M}, cost, goal)$, wobei:

1. Σ_I ist ein Alphabet (das **Eingabealphabet**)
2. Σ_O ist ein Alphabet (das **Ausgabealphabet**)
3. $L \subseteq \Sigma_I^*$ ist die Sprache der **zulässigen Eingaben**. Ein Wörter in L heißen **Instanzen** (**Problemfälle**).
4. $\mathcal{M}$ ist eine Funktion von L nach $\mathcal{P}(\Sigma_O^*)$. Für $x \in L$ ist $\mathcal{M}(x)$ die **Menge der zulässigen Lösungen für** x.
5. $cost\colon \bigcup_{x \in L}(\mathcal{M}(x), x) \to \mathbf{R}^+$ ist eine Funktion, die **Preisfunktion**.
6. $goal \in \{Minimum, Maximum\}$ ist das **Optimierungsziel**.

Definition 2.13 (Fortsetzung)

Eine zulässige Lösung $\alpha \in \mathcal{M}(x)$ heißt **optimal** für den Problemfall x des Optimierungsproblems $\mathcal{U}$, falls

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

Ein Algorithmus A **löst** U, falls für jedes $x \in L$

1. $A(x) \in \mathcal{M}(x)$ (Lösung ist zulässig)
2. $cost(A(x)) = \mathrm{Opt}_{\mathcal{U}}(x)$ (Lösung ist optimal).

Je nach *goal* sprechen wir von einem **Minimierungs-** oder einem **Maximierungsproblem**.

Definition 2.16

Für jedes Wort $x \in \{0,1\}^*$ ist die **Kolmogorov-Komplexität** $K(x)$ des Wortes x die binäre Länge des kürzesten Pascal-Programms, das x generiert.

Definition 2.17

Die **Kolmogorov-Komplexität** einer Zahl n ist $K(n) = K(Bin(n))$.

Definition 3.1

Ein (deterministischer) **endlicher Automat (EA)** ist ein Quintupel $M = (Q, \Sigma, \delta, q_0, F)$, wobei

1. Q eine endliche Menge von **Zuständen** ist,
2. Σ ein Alphabet, das **Eingabealphabet** ist,
3. $q_0 \in Q$ der **Anfangszustand** ist,
4. $F \in Q$ die Menge der **akzeptierenden Zustände (Endzustände)** ist,
5. $\delta\colon Q \times \Sigma \to Q$ die **Übergangsfunktion** ist.

Definition 3.1 (Fortsetzung)

Eine **Konfiguration** C von M ist ein Element aus $Q \times \Sigma^*$. (Wenn M sich in der Konfiguration (q, w) befindet, ist er im Zustand q und muß noch den Suffix w der Eingabe lesen.)

Eine Konfiguration (q_0, x) ist die **Startkonfiguration von** M **auf** x.

Die Konfigurationen aus $Q \times \{\lambda\}$ nennt man **Endkonfigurationen**.

Die **Schrittrelation** von M ist $\underset{M}{\vdash} \subseteq (Q \times \Sigma^*) \times (Q \times \Sigma^*)$, definiert durch

$(q, w) \underset{M}{\vdash} (p, x) \iff w = ax,\ a \in \Sigma$ und $\delta(q, a) = p$.

Definition 3.1 (Fortsetzung)

Eine **Berechnung** C von M ist eine endliche Folge $C = (C_0, C_1, \ldots, C_n)$ von Konfigurationen, so daß $C_i \underset{M}{\vdash} C_{i+1}$ für alle $0 \leq i < n$.

Wir sagen C sei **eine Berechnung von** M **auf einer Eingabe** $x \in \Sigma^*$, falls $C_0 = (q_0, x)$ und $C_n \in Q \times \{\lambda\}$ eine Endkonfiguration ist.

Falls $C_n \in F \times \{\lambda\}$, dann ist C eine **akzeptierende** Berechnung von M auf x und M **akzeptiert das Wort** x.

Falls $C_n \in (Q - F) \times \{\lambda\}$, dann ist C eine **verwerfende** Berechnung von M auf x und M **verwirft das Wort** x.

Definition 3.1 (Fortsetzung)

Die von M **akzeptierte Sprache** $L(M)$ ist definiert als

$L(M) := \{\, w \in \Sigma^* \mid$ die Berechnung von M auf w endet in einer Endkonfiguration (q, λ) mit $q \in F \,\}$

$\mathcal{L}(EA) = \{\, L(M) \mid M$ ist ein EA $\,\}$ ist die Klasse der Sprachen, die von endlichen Automaten akzeptiert werden.

Sie werden auch als **reguläre Sprachen** bezeichnet.

Definition 3.2

Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein endlicher Automat. Wir definieren $\overset{*}{\underset{M}{\vdash}}$ als die reflexive und transitive Hülle der Schrittrelation $\underset{M}{\vdash}$ von M.

Daher gilt $(q, w) \overset{*}{\underset{M}{\vdash}} (p, u)$ genau dann, wenn

$(q, w) = (r_0, a_1 a_2 \ldots a_n u) \underset{M}{\vdash} (r_1, a_2 \ldots a_n u) \underset{M}{\vdash} \cdots \underset{M}{\vdash} (r_{n-1}, a_n u) \underset{M}{\vdash} (r_n, u) = (p, u)$.

Wir definieren $\hat{\delta} \colon Q \times \Sigma^* \to Q$ durch

1. $\hat{\delta}(q, \lambda) = q$ für alle $q \in Q$ und
2. $\hat{\delta}(q, wa) = \delta(\hat{\delta}(q, w), a)$ für alle $a \in \Sigma$, $w \in \Sigma^*$, $q \in Q$.

Definition 4.1

Eine **Turingmaschine** ist ein 7-Tupel
$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$. Dabei ist

1. Q eine endliche Menge, die **Zustandsmenge**,
2. Σ das **Eingabealphabet**, welches $\not\in$ und $\sqcup$ nicht enthält,
3. Γ das Arbeitsbandalphabet, mit $\Sigma \subseteq \Gamma$ und $\not\in, \sqcup \in \Gamma$,
4. $\delta\colon (Q - \{q_{accept}, q_{reject}\}) \times \Gamma \to Q \times \Gamma \times \{L, R, N\}$ eine Abbildung, die **Übergangsfunktion**, mit $\delta(q, \not\in) \in Q \times \{\not\in\} \times \{R, N\}$ für alle $q \in Q$.
5. $q_0 \in Q$ der **Anfangszustand**,
6. $q_{accept} \in Q$ der **akzeptierende Zustand**,
7. $q_{reject} \in Q$ der **verwerfende Zustand**.

Definition 4.1 (Fortsetzung)

Eine **Konfiguration** C von M ist ein Element aus

$$\mathrm{Konf}(M) = \{\not\in\}\Gamma^* Q \Gamma^+ \cup Q\{\not\in\}\Gamma^*.$$

Eine **Startkonfiguration** für ein Eingabewort x ist $q_0 \not\in x$.

Die Schrittrelation $\underset{M}{\vdash} \subseteq \mathrm{Konf}(M) \times \mathrm{Konf}(M)$ ist definiert durch

1. $x_1 \ldots x_{i-1} q x_i \ldots x_n \underset{M}{\vdash} x_1 \ldots x_{i-1} p y x_{i+1} \ldots x_n$ falls $\delta(q, x_i) = (p, y, N)$ (oder $\delta(q, \sqcup) = (p, y, N)$ und $n = i - 1$),
2. $x_1 \ldots x_{i-1} q x_i \ldots x_n \underset{M}{\vdash} x_1 \ldots x_{i-2} p x_{i-1} y x_{i+1} \ldots x_n$ falls $\delta(q, x_i) = (p, y, L)$ (oder $\delta(q, \sqcup) = (p, y, L)$ und $n = i - 1$),
3. $x_1 \ldots x_{i-1} q x_i \ldots x_n \underset{M}{\vdash} x_1 \ldots x_{i-1} y p x_{i+1} \ldots x_n$ falls $\delta(q, x_i) = (p, y, R)$ (oder $\delta(q, \sqcup) = (p, y, R)$ und $n = i - 1$).

Definition 4.1 (Fortsetzung)

Eine **Berechnung** von M ist eine (potentiell unendliche) Folge von Konfigurationen $C_0, C_1, C_2, \ldots$, so daß $C_i \underset{M}{\vdash} C_{i+1}$ für alle i.

Die reflexiv-transitive Hülle von $\underset{M}{\vdash}$ ist wieder $\overset{*}{\underset{M}{\vdash}}$.

Eine **Berechnung von M auf einer Eingabe** x ist eine Berechnung, die mit der Startkonfiguration $C_0 = q_0 \not\in x$ beginnt und entweder unendlich ist oder in einer Konfiguration $w_1 q w_2$ endet, mit $q \in \{q_{accept}, q_{reject}\}$.

Eine Berechnung von M auf x heißt **akzeptierend**, falls sie in einer **akzeptierenden Konfiguration** $w_1 q_{accept} w_2$ endet. Eine Berechnung von M auf x heißt **verwerfend**, falls sie in einer **verwerfenden Konfiguration** $w_1 q_{reject} w_2$ endet. Eine **nicht-akzeptierende** Berechnung von M auf x ist eine verwerfende oder unendliche Berechnung von M auf x.

Definition 4.1 (Fortsetzung)

Die von der **Turingmaschine** M **akzeptierte Sprache** ist

$$L(M) = \{\, w \in \Sigma^* \mid q_0 \not\in w \overset{*}{\underset{M}{\vdash}} y q_{accept} z \text{ für } y, z \in \Gamma^* \,\}.$$

Eine Sprache heißt **rekursiv aufzählbar**, falls eine TM M existiert mit $L = L(M)$.

$\mathcal{L}_{\mathrm{RE}} = \{\, L(M) \mid M \text{ ist eine TM} \,\}$ ist die Klasse der **rekursiv aufzählbaren Sprachen**.

Eine Sprache heißt **rekursiv** oder **entscheidbar**, falls $L = L(M)$ für eine TM M, die auf allen Eingaben $x \in \Sigma^*$ hält.

$\mathcal{L}_{\mathrm{R}} = \{\, L(M) \mid M \text{ ist eine TM, die immer hält} \,\}$ ist die Klasse der **rekursiven (entscheidbaren) Sprachen**.

Definition 5.3

Es seien $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ zwei Sprachen. Wir sagen, daß L_1 **auf** L_2 **rekursiv reduzierbar** ist, in Zeichen $L_1 \leq_R L_2$, falls

$L_2 \in \mathcal{L}_R \Rightarrow L_1 \in \mathcal{L}_R$.

Definition 5.4

Es seien $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ zwei Sprachen. Wir sagen, daß L_1 **auf** L_2 **EE-reduzierbar** ist, in Zeichen $L_1 \leq_{EE} L_2$, falls eine TM M existiert, die eine Abbildung $f_M \colon \Sigma_1^* \to \Sigma_2^*$ mit der Eigenschaft

$x \in L_1 \iff f_M(x) \in L_2$

für alle $x \in \Sigma_1^*$ berechnet. Wir sagen, daß M die **Sprache** L_1 **auf die Sprache** L_2 **reduziert**.