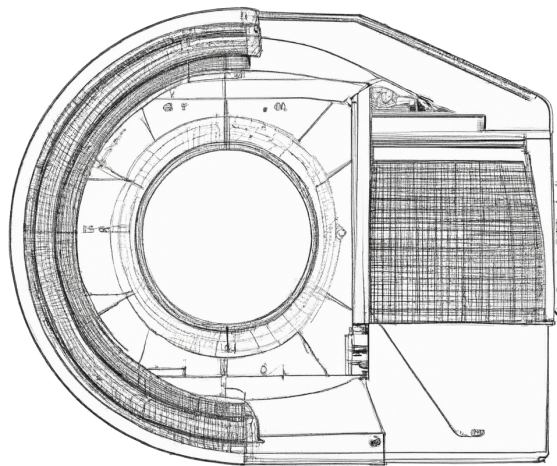


Einführung in die Theoretische Informatik

Felix Ichtters*

Sommersemester 2023

Begleitmaterial zur Vorlesung 'Einführung in die Theoretische Informatik'.



*Universität Heidelberg

Inhaltsverzeichnis

1	Grundlagen	5
1.1	Alphabet	6
1.2	Wörter	6
1.3	Sprache	6
1.4	Σ^*	6
1.5	Verkettung	6
1.6	Präfix, Infix, Suffix	6
1.7	Homomorphismus	6
1.8	Längenlexikographische Ordnungen	7
1.9	$\text{bin}(i)$	7
2	Turingmaschinen	8
2.1	Turingmaschine	9
2.2	Konfiguration	9
2.3	Nachfolgekonfiguration	9
2.4	Rechnung	10
2.5	Total	10
2.6	Akzeptierte Sprache	10
2.7	Entscheidbar	10
2.8	Rekursiv aufzählbar	10
2.9	Ausgabe	11
2.10	Berechnete Funktion	11
2.11	Partiell Berechnbar	11
2.12	Charakteristische Funktion	11
2.13	Normiert	12
2.14	Code	12
2.15	Standardaufzählung	13
2.16	U	13
2.17	Universell	13
2.18	$s \leq_m n$ -Theorem	13
2.19	Diagonales Halteproblem	14
2.20	m -Reduktion	14
2.21	Postisches Korrespondenzproblem	14
2.22	Fixpunkt	14
2.23	Rekursionstheorem	14
2.24	Indexmenge	15
3	Automaten	16
3.1	Endlicher Automat	17
3.2	Übergangsfunktion eines EA	17
3.3	Übergangsfunktion eines DEA	17
3.4	akzeptierte Sprache	18
3.5	Regulär	18
3.6	Übergangsdiagramm	18
3.7	Potenzautomat	18

4	Reguläre Sprachen	20
4.1	Äquivalenzrelation	21
4.2	A-Äquivalenz	21
4.3	Rechtskongurenz	21
4.5	Erreichbar	22
4.6	Isomorph	23
4.7	L-Äquivalenz	23
4.8	Partition	24
4.9	Verfeinerung	24
4.10	Minimalautomat	24

Notationen

- Für $n \in \mathbb{N}_0$, sei $[n] = \{1, \dots, n\}$ und $[n]_0 = \{0, 1, \dots, n\}$.
- Für eine Menge A und $n \in \mathbb{N}$ ist $A^n = \{(a_1, \dots, a_n) : a_1, \dots, a_n \in A\}$.
- Für $n \in \mathbb{N}$ ist eine n -äre partielle Funktion $\varphi : A^n \rightsquigarrow B$ eine Funktion mit $\text{dom}(\varphi) \subseteq A^n$ und $\text{Im}(\varphi) \subseteq B$. Für $a_1, \dots, a_n \in A$ bedeutet $\varphi(a_1, \dots, a_n) \downarrow$, dass $a_1, \dots, a_n \in \text{dom}(\varphi)$ gilt und $\varphi(a_1, \dots, a_n) \uparrow$ bedeutet, dass $(a_1, \dots, a_n) \notin \text{dom}(\varphi)$. Die partielle Funktion φ ist total, wenn $\text{dom}(\varphi) = A^n$ gilt.
- Eine lineare Ordnung auch totale Ordnung, auf einer Menge A ist eine Relation $\leq \subseteq A^2$, so dass die folgenden Eigenschaften erfüllt sind

$$a \leq a \quad \forall a \in A \quad (\text{Reflexivität})$$

$$a \leq b \wedge b \leq a \Rightarrow a = b \quad \forall a, b \in A \quad (\text{Antisymmetrie})$$

$$a \leq b, b \leq c \Rightarrow a \leq c \quad \forall a, b, c \in A \quad (\text{Transitivität})$$

$$a \leq b \vee b \leq a \quad \forall a, b \in A \quad (\text{Totalität})$$

1 Grundlagen

1.1 Alphabet

Ein Alphabet ist eine nichtleere Menge Σ , die Elemente heißen Symbole.

1.2 Wörter

Ein Wort über einem Alphabet Σ ist eine endliche Folge von Symbolen aus Σ .

Für $i \in [|w|]$ bezeichnet $w(i)$ das i -te Element von w und für Symbole $a_1, \dots, a_n \in \Sigma$ bezeichnet a_1, a_2, \dots, a_n das Wort w der Länge n mit $w(i) = a_i \forall i \in [n]$.

1.3 Sprache

Eine Sprache ist eine Menge von Wörtern über einem gemeinsamen Alphabet Σ .

1.4 Σ^*

Die Menge aller Wörter über Σ wird mit Σ^* bezeichnet.

Für $n \in \mathbb{N}_0$ setzen wir

$$\Sigma^{\leq n} := \{w \in \Sigma^* : |w| \leq n\}$$

$$\Sigma^{=n} := \{w \in \Sigma^* : |w| = n\}$$

$$\Sigma^{\geq n} := \{w \in \Sigma^* : |w| \geq n\}$$

$$\Sigma^+ := \Sigma^{\geq 1}$$

1.5 Verkettung

Für Wörter w_1, w_2 ist die Verkettung $w_1 w_2$ ist definiert durch

$$w_1 w_2 := w_1(1) \dots w_1(|w_1|) w_2(1) \dots w_2(|w_2|).$$

Für Sprachen L_1, L_2 sei $L_1 L_2$ definiert durch

$$L_1 L_2 := \{w_1 w_2 : w_1 \in L_1, w_2 \in L_2\}$$

1.6 Präfix, Infix, Suffix

Seien u, v Wörter

- u ist Präfix von v , kurz $u \sqsubseteq v$, falls es ein Wort w gibt, sodass $uw = v$
- u ist Infix von v , falls es Wörter w_1, w_2 gibt, sodass $v = w_1 u w_2$
- u ist Suffix von v , falls es ein Wort w gibt, sodass $v = wu$

1.7 Homomorphismus

Für Sprachen L, M heißt eine Funktion $\varphi : L \rightarrow M$ Homomorphismus von Sprachen, wenn gilt:

$$\varphi(uv) = \varphi(u)\varphi(v)$$

1.8 Längenlexikographische Ordnungen

Es gilt $u \leq_{lex} v$ wenn eine der folgenden Bedingungen erfüllt ist:

1. $|u| < |v|$
2. $|u| = |v|$ und ist $i \in [|u|]$ minimal mit $u(i) \neq v(i)$, so gilt $u(i) \leq v(i)$

1.9 bin(i)

$bin(i)$ ist die Funktion für das in Längenlexikographischer Reihenfolge $i + 1$ -te Binärwort.
 $1bin(i)$ beschreibt die Binärdarstellung von $i + 1$.

2 Turingmaschinen

2.1 Turingmaschine

Eine k-Band Turingmaschine ist ein Tupel

$$M = (Q, \Sigma, \Gamma, \Delta, s, F)$$

- Q ist die endliche Zustandsmenge
- Σ ist das Eingabealphabet
- Γ das Bandalphabet mit $\Sigma \subseteq \Gamma$ und $\square \in \Gamma \setminus \Sigma$
- $\Delta \subseteq Q \times \Gamma^k \times Q \times \Gamma^k \times \{L, S, R\}^k$ die Übergangsrelation
- $s \in Q$ der Startzustand
- $F \subseteq Q$ die Menge der akzeptierten Zustände

Die Elemente von Δ heißen Instruktionen, eine Instruktion sieht wie folgt aus:

$$(q, a_1, \dots, a_k, q', a'_1, \dots, a'_k, B_1, \dots, B_k)$$

Eine TM ist eine DTM, wenn es $\forall b \in Q \times \Gamma^k$ höchstens eine Instruktion $i \in \Delta$ mit Bedingungsteil b gibt

2.2 Konfiguration

Eine Konfiguration einer k-TM ist ein Tupel

$$C = (q, w_1, \dots, w_k, p_1, \dots, p_k) \in Q \times (\Gamma^*)^k \times \mathbb{N}^k$$

Die Startkonfiguration zur Eingabe $(u_1, \dots, u_n) \in (\Sigma^*)^n$ mit $n \in \mathbb{N}$, ist die Konfiguration

$$\text{start}(u_1, \dots, u_n) = (s, u_1 \square u_2 \square \dots \square u_n, \square, \dots, \square, 1, \dots, 1)$$

Die Stoppkonfiguration ist die Konfiguration zu der es keine Nachfolgekongfiguration gibt

2.3 Nachfolgekongfiguration

Für Konfigurationen $C = (q, w_1, \dots, w_k, p_1, \dots, p_k)$ und $C' = (q', w'_1, \dots, w'_k, p'_1, \dots, p'_k)$ einer k-TM,

ist die Konfiguration C' **Nachfolgekongfiguration** von C , wenn es eine Instruktion

$$(q, w_1(p_1), \dots, w_k(p_k), q', a'_1, \dots, a'_k, B_1, \dots, B_k) \in \Delta$$

gibt sodass

$$w_i = \begin{cases} \square a'_i w_i(2) \dots w_i(|w_i|) & \text{falls } p_i = 1 \text{ und } B_i = L \\ w_i(1) \dots w_i(|w_i| - 1) a'_i \square & \text{falls } p_i = |w_i| \text{ und } B_i = R \\ w_i(1) \dots w_i(p_i - 1) a'_i w_i(p_i + 1) \dots w_i(|w_i|) & \text{sonst} \end{cases}$$

und

$$w_i = \begin{cases} 1 & \text{falls } p_i = 1 \text{ und } B_i = L \\ p_i - 1 & \text{falls } p_i \geq 2 \text{ und } B_i = L \\ p_i & \text{falls } B_i = S \\ p_i + 1 & \text{falls } B_i = R \end{cases}$$

für alle $i \in [k]$ gelten.

2.4 Rechnung

Es bezeichne \rightarrow_M die Relation auf der Menge der Konfigurationen einer k -TM M , sodass $C \rightarrow_M C'$ falls C, C' Konfigurationen von M sind wobei C' eine Nachfolgekongfiguration von C ist.

Eine **endliche partielle Rechnung** ist eine endliche Folge C_1, \dots, C_n von Konfigurationen von M mit $C_i \rightarrow_M C_{i+1} \forall i \in [n-1]$.

Eine **unendlich partielle Rechnung** ist eine unendliche Folge C_1, C_2, \dots von Konfigurationen von M mit $C_i \rightarrow_M C_{i+1} \forall i+1 \in \mathbb{N}$. Eine Rechnung zur Eingabe $(w_1, \dots, w_n) \in (\Sigma^*)^n$ mit $n \in \mathbb{N}$ ist ein unendlich partielle Rechnung $start_M(w_1, \dots, w_n) = C_1, C_2, \dots, C_m$.

Ist M eine k -DTM, so gilt es $\forall n \in \mathbb{N}$ und $(w_1, \dots, w_n) \in (\Sigma^*)^n$ genau eine Rechnung zur Eingabe (w_1, \dots, w_n) . Ist M eine k -DTM, so gilt es $\forall n \in \mathbb{N}$ und $(w_1, \dots, w_n) \in (\Sigma^*)^n$ genau eine Rechnung zur Eingabe w_1, \dots, w_n .

2.5 Total

Totale TMs, sind TMs die bei jeder Eingabe immer anhalten. Alle Rechnungen müssen endlich sein.

2.6 Akzeptierte Sprache

Eine Stoppkonfiguration ist akzeptiert, wenn $q \in F$.

Die akzeptierte Sprache $L(M)$ ist die Sprache über dem Alphabet Σ , so dass $\forall w \in \Sigma^*$ genau dann $w \in L(M)$ gilt, wenn es eine endliche Rechnung C_1, \dots, C_n zur Eingabe w gibt, bei der C_n eine akzeptierte Stoppkonfiguration ist.

Für nicht-deterministische Tms heißt das, dass es für die Wörter in der akzeptierten Sprache nur mind. eine in einer akzeptierten Stoppkonfiguration endende endliche Rechnungen zur Eingabe w geben muss.

Für Wörter w die nicht in $L(M)$ sind, sind alle Rechnung von M zur Eingabe am Ende nicht in einer akzeptierten Stoppkonfiguration oder unendlich

2.7 Entscheidbar

Eine Sprache L ist genau dann entscheidbar, wenn es eine totale k -TM mit $L(M) = L$ gibt.

2.8 Rekursiv aufzählbar

Eine Sprache L ist genau dann rekursiv aufzählbar, wenn es eine k -TM mit akzeptierter Sprache L gibt.

Wir schreiben RE für die Klasse der rekursiv aufzählbaren Sprachen. Die Aufzählbarkeit leitet sich daraus ab, dass es für eine rekursiv aufzählbare Sprache L über einem Alphabet Σ möglich ist effektive Verfahren anzugeben, die die Wörter von L aufzählen, also dass eine endlich oder unendliche Aufzählung von $A = w_1, w_2, \dots$ mit $w_1, w_2, \dots = L$ existiert.

- Jede entscheidbare Sprache ist rekursiv aufzählbar

- Alle endlichen Sprachen sind entscheidbar
- Eine Sprache L über einem Alphabet Σ ist genau dann entscheidbar, wenn L und $L^c := (\Sigma^*)/L$ rekursiv aufzählbar sind.

2.9 Ausgabe

Die Ausgabe $out_M(C)$ bei Konfiguration C ist das Präfix $w \sqsubseteq w_1(p_1) \dots w_1(|w_1|)$ max. Länge mit $w \in (\Gamma \setminus \{\square\})^*$.

Also das längst mögliche Wort, dass auf Band 1 rechts vom Lesekopf steht und nicht \square enthält.

2.10 Berechnete Funktion

Sei M eine k -DTM. Die von M berechnete n -äre partielle Funktion φ_M ist die partielle Funktion:

$$\varphi_M : (\Sigma^*)^n \rightsquigarrow (\Gamma \setminus \{\square\})^*$$

sodass folgendes gilt:

1. Ist die Rechnung zur Eingabe w_1, \dots, w_n die endliche Rechnung C_1, \dots, C_m so gilt $\varphi_M(w_1, \dots, w_n) = out_M(C_m)$
2. Ist die Rechnung zur Eingabe w_1, \dots, w_n unendlich, so gilt $\varphi_M(w_1, \dots, w_n) \uparrow$

Nur deterministische TMs können Funktionen berechnen.

2.11 Partiiell Berechnbar

Für Σ, Γ und eine partielle Funktion $U : \Sigma^* \rightsquigarrow \Gamma^*$ ist φ berechenbar, wenn es ein $k \in \mathbb{N}$ gibt und eine k -DTM M mit $\varphi_M = \varphi$ gibt.

Ist φ total und partiell berechenbar, so ist φ berechenbar.

2.12 Charakteristische Funktion

Sei L eine Sprache über dem Alphabet Σ .

1. Die charakteristische Funktion von L als Sprache über Σ ist die Funktion $\mathbb{1}_L : \Sigma^* \rightarrow \{0, 1\}$ mit $\mathbb{1}_L(w) = 1 \ \forall w \in L$ und $\mathbb{1}_L(w) = 0 \ \forall w \in \Sigma^* \setminus L$.
2. Die partiell charakteristische Funktion von L als Sprache über Σ ist die partielle Funktion $\chi_L : \Sigma^* \rightsquigarrow \{1\}$ mit $\chi_L(w) = 1 \ \forall w \in L$ und $\chi_L(w) \uparrow \ \forall w \in \Sigma^* \setminus L$.

Bemerkung

Sei L eine Sprache über einem Alphabet Σ

- L ist genau dann entscheidbar, wenn $\mathbb{1}_L$ berechenbar ist.
- L ist genau dann rekursiv aufzählbar, wenn χ_L partiell berechenbar ist.

2.13 Normiet

Eine 1-DTM M heißt normiert, wenn $Q=0,\dots,n$ für ein $n \in \mathbb{N}_0$, $\Sigma = \{0,1\}$, $\Gamma = \{\square, 0,1\}$, $\Delta = 0$, $F = \{s\}$.

Alle TMs mit Eingabealphabet $0,1$ lassen sich mit folgenden Schritten in eine normierte TM mit gleicher erkannter Sprache und gleicher berechneten Funktion umwandeln.

- Von nicht-Determinismus zu Determinismus

Eine DTM kann die Rechnungen einer nicht-Deterministischen TM parallel im Sinne von abwechselnd schrittweise durchführen um schließlich das Verhalten der simulierten TM zu imitieren. Das entspricht einer Breitensuche im Rechnungsbaum.

- Von mehreren Bändern zu einem Band

Intuitiv können k Bänder auf einem Band simuliert werden, indem die Felder des einen Bandes in k -Teilfelder unterteilt werden, die jeweils die gleichen Bandalphabetbuchstaben wie zuvor als Beschriftung zulassen und es zudem erlauben zu notieren, dass der simulierte Kopf des simulierten Bandes dort steht.

- Von beliebigem Bandalphabet zu $\{\square, 0,1\}$

Andere Bandalphabete können bei einem Alphabet Wechsel zum Bandalphabet $\{\square, 0,1\}$ simuliert werden, indem mehrere nebeneinander liegende Felder verwendet werden um ein Symbol des vorigen Bandalphabets durch ein Binärwort zu beschreiben. Die TM liest stets nur ein Feld, es wird daher also nötig sein die Zustandsmenge so zu erweitern, dass angrenzende Felder im Zustand gespeichert werden können.

2.14 Code

Wir betrachten die Funktion $code$ mit geeigneter Definitionsmenge und Zielmenge $\{0,1\}$.

- Codieren der Bewegungsrichtung

$$code(L) = 10$$

$$code(S) = 00$$

$$code(R) = 01$$

- Codieren der einzelnen Instruktionen

$$I = (q, a, q', a', B)$$

$$code(I) = 0^{|\text{bin}(q)|} 1 \text{bin}(q) a 0^{|\text{bin}(q')|} 1 \text{bin}(q') a' code(B)$$

- Codieren des Instruktionssatzes

$$code(\Delta) = code_1(\Delta) \dots code_{|\Delta|}(\Delta)$$

- Codieren der normierten TM

$$code(M) = 0^{|\text{bin}(n)|} 1 \text{bin}(n) code(\Delta)$$

Jede normierte TM hat einen Code und zwei verschiedene niemals den gleichen. Die Sprache der TMs ist entscheidbar.

2.15 Standardaufzählung

Sei $\hat{w}_0, \hat{w}_1, \dots$ die Aufzählung aller Codes normierter TMs in längenlexikographischer Ordnung. Für $e \in \mathbb{N}$ sei \mathcal{M}_e die durch \hat{w}_e codierte TM und für $n \in \mathbb{N}$ sei $\Phi_e^n : \mathbb{N}_0^n \rightarrow \mathbb{N}_0$ die von \mathcal{M}_e

berechnete n -äre partielle Funktion.

Für $n \in \mathbb{N}$ heißt die Folge $(\Phi_e^n)_e \in \mathbb{N}$ Standardaufzählung der n -ären partiell berechenbaren Funktion.

Für $n \in \mathbb{N}$ und eine partiell berechenbare n -äre partielle Funktion $\varphi : \mathbb{N}_0^n \rightarrow \mathbb{N}_0$ heißt jede Zahl $e \in \mathbb{N}_0$ mit $\Phi_e^n = \varphi$ Index von φ .

Ergibt sich n aus dem Kontext so schreiben wir auch Φ_e statt Φ_e^n

Für $n \in \mathbb{N}$ und eine partielle berechnbare n -äre partielle Funktion $\Phi : \mathbb{N}_0^n \rightarrow \mathbb{N}_0$ gibt es unendlich viele Indizes von Φ .

2.16 U

Es bezeichne \mathcal{U} die normierte TM, bei Eingabe $(e, x_1, \dots, x_n) \in \mathbb{N}_0^{n+1}$ wobei $n \in \mathbb{N}$ die normierte TM \mathcal{M}_e bei Eingabe x_1, \dots, x_n simuliert und falls diese terminiert die Ausgabe der Simulation ausgibt.

2.17 Universell

Eine DTM \mathcal{U} heißt universell, wenn es für alle $n \in \mathbb{N}$ und alle partiell berechenbaren Funktionen $\varphi : \mathbb{N}_0^n \rightarrow \mathbb{N}_0$ ein $e \in \mathbb{N}_0$ gibt, sodass $\mathcal{U}(e, x_1, \dots, x_n) = \varphi(x_1, \dots, x_n)$ für alle $x_1, \dots, x_n \in \mathbb{N}_0$ gilt.

2.18 s m n-Theorem

Für alle $m, n \in \mathbb{N}$ existiert eine berechenbare Funktion $s_n^m : \mathbb{N}_0^{m+1} \rightarrow \mathbb{N}_0$ mit

$$\Phi_e^{m+n}(x_1, \dots, x_m, y_1, \dots, y_n) = \Phi_{s_n^m(e, x_1, \dots, x_m)}^n(y_1, \dots, y_n)$$

für alle $e, x_1, \dots, x_m, y_1, \dots, y_n \in \mathbb{N}_0$.

Beweis. Fixiere $m \in \mathbb{N}$. Betrachte die DTM S , die bei Eingabe $(e, x_1, \dots, x_m) \in \mathbb{N}_0^{m+1}$ wie folgt verfährt?.

- Zunächst bestimmt S den Code von \mathcal{M}_e
- der Code von \mathcal{M}_1 wird dann in einen Code einer normierten TM \mathcal{M} umgewandelt, die zunächst $x_1 \square \dots \square x_m \square$ neben die Eingabe schreibt, dann den Kopf auf das erste Feld des beschriebenen Bandteilsbewegt und dann wie \mathcal{M}_1 arbeitet.
- Es wird bestimmt an welcher Stelle der Standardaufzählung der Code von auftaucht und diese Stelle wird ausgegeben.

Sei s_n^m die von S berechnete $(m+1)$ -äre partielle Funktion. Dann ist s_n^m eine Funktion wie gewünscht. \square

2.19 Diagonales Halteproblem

Die Menge $H_{diag} := \{e \in \mathbb{N}_0 : \Phi_e(e) \downarrow\}$ heißt diagonales Halteproblem.

Das diagonale Halteproblem ist rekursiv aufzählbar, jedoch nicht entscheidbar.

2.20 m-Reduktion

Für eine Sprache A über einem Alphabet Σ und eine Sprache B über einem Alphabet Γ ist A genau dann many-one-reduzierbar, auch m-reduzierbar, auf B , kurz $A \leq_m B$, wenn es eine berechenbare Funktion $f : \Sigma^* \rightarrow \Gamma^*$ gibt, so dass

$$w \in A \Leftrightarrow f(w) \in B$$

für alle $w \in \Sigma^*$ gilt.

gelten $A \leq_m B$ und $B \leq_m A$, so sind A und B m-äquivalent, kurz $A =_m B$.

2.21 Postsches Korrespondenzproblem

Für ein Alphabet Σ sei eine Instanz des Postschen Korrespondenzproblem über Σ eine endliche Teilmenge $I \subseteq (\Sigma^+)^2$. Eine Lösung für eine solche Instanz ist eine endliche Folge $(u_1, v_1), \dots, (u_n, v_n)$ von Paaren in I mit $n \geq 1$, so dass

$$u_1 \dots u_n = v_1 \dots v_n$$

Gibt es eine Lösung für eine Instanz des Postschen Korrespondenzproblems, so heißt diese Instanz lösbar.

Das Postsche Korrespondenzproblem über einem Alphabet Σ , kurz PCP_Σ , ist die Menge aller lösbarer Instanzen des Postschen Korrespondenzproblems über Σ .

Für ein Alphabet Σ sei eine Instanz des modifizierten Postschen Korrespondenzproblems über Σ ein Paar (p, I) , wobei $I \subseteq (\Sigma^+)^2$ eine endliche Teilmenge und $p \in I$ ein Paar von Wörtern ist.

Eine Lösung für eine solche Instanz ist eine endliche Folge $(u_1, v_1), \dots, (u_n, v_n)$ von Paaren in I , so dass

$$p = (u_1, v_1) \text{ und } u_1 \dots u_n = v_1 \dots v_n$$

Gibt es eine Lösung für eine Instanz des modifizierten Postschen Korrespondenzproblems, so heißt diese Instanz lösbar.

Das modifizierte Postsche Korrespondenzproblem über einem Alphabet Σ , kurz $MPCP_\Sigma$ ist die Menge aller lösbarer Instanzen des modifizierten Postschen Korrespondenzproblems über Σ .

2.22 Fixpunkt

Ein Fixpunkt einer berechenbaren Funktion $f : \mathbb{N}_0 \rightarrow \mathbb{N}_0$ ist ein $e \in \mathbb{N}_0$ mit $\Phi_{f(e)} = \Phi_e$.

2.23 Rekursionstheorem

Für alle partiell berechenbaren Funktion $\varphi : \mathbb{N}_0^2 \rightsquigarrow \mathbb{N}_0$ gibt es ein $e \in \mathbb{N}_0$ mit $\Phi_e(x) = \varphi(e, x) \forall x \in \mathbb{N}_0$.

2.24 Indexmenge

Eine Teilmenge $I \subseteq \mathbb{N}_0$ heißt Indexmenge, wenn $e \in I \Leftrightarrow e' \in I$ für alle $e, e' \in \mathbb{N}_0$ mit $\Phi_e = \Phi_{e'}$ gilt.

3 Automaten

3.1 Endlicher Automat

Ein endlicher Automat, EA, ist ein Tupel $A = (Q, \Sigma, \Delta, s, F)$.

- Q ist eine endliche Menge, die Zustandsmenge
- Σ das Eingabealphabet
- $\Delta \subseteq Q \times \Sigma \times Q$ die Übergangsrelation, so dass es für alle $q \in Q$ und $a \in \Sigma$ ein $q' \in Q$ mit $(q, a, q') \in \Delta$
- $s \in Q$ der Startzustand
- $F \subseteq Q$ die Menge der akzeptierenden Zustände

Der endliche Automat A ist ein deterministischer endlicher Automat, kurz DEA, wenn es $\forall (q, a) \in Q \times \Sigma$ genau ein q' gibt mit $(q, a, q') \in \Delta$. Im Sinne der obigen Betrachtung entspricht ein EA $A = (Q, \Sigma, \Delta, s, F)$ der 1-TM $M_A = (Q, \Sigma, \Sigma \cup \{\square\}, \{(q, a, q', a, R) : (q, a, q') \in \Delta\}, s, F)$.

3.2 Übergangsfunktion eines EA

Die Übergangsfunktion eines EA A ist die Funktion $\delta_A^*(q, \lambda) = \{q\}$ und

$$\delta_A^*(q, aw) = \bigcup_{q' \in \delta_A(q, a)} \delta_A^*(q', w)$$

$\forall q \in Q, a \in \Sigma, w \in \Sigma^*$

Für $Q_0 \subseteq Q$ und $w \in \Sigma^*$ schreiben wir $\delta_A^*(Q_0, w)$ statt $\bigcup_{q \in Q_0} \delta_A^*(q, w)$.

Sei $A = (Q, \Sigma, \Delta, s, F)$ ein EA

(i) $\forall q \in Q$ und $a \in \Sigma$ gilt $\delta_A^*(q, a) = \delta_A(q, a)$.

(ii) Ist A ein DEA, $q \in Q$, $a \in \Sigma$ und $w \in \Sigma^*$, und $|\delta_A^*(q, w)| = 1$??.

(iii) Seien $u, v \in \Sigma^*$ $\forall q \in Q$ gilt $\delta_A^*(q, uv) = \delta_A^*(\delta_A^*(Q_0, u), v)$.

3.3 Übergangsfunktion eines DEA

Sei A ein DEA. Auch die Funktion $\delta_{det, A} : Q \times \Sigma \rightarrow Q$ mit $\delta_a(q, a) = \{\delta_{det, A}(q, a)\} \forall q \in Q$ und $a \in \Sigma$ wird Übergangsfunktion von A genannt. Analoges gilt für $\delta_{det, A}^*$ und δ_A^* .

Für $Q_0 \subseteq Q$ und $w \in \Sigma^*$ schreiben wir auch $\delta_{det, A}^*(Q_0, w)$.

Sei Q eine endliche Menge, Σ ein Alphabet, $s \in Q$, und $F \subseteq Q$.

(i) \forall Funktionen $\delta : Q \times \Sigma \rightarrow 2^Q$ gibt es genau einen EA $A = (Q, \Sigma, \Delta, s, F)$ mit $\delta_A = \delta$.

(ii) \forall Funktionen $\delta : Q \times \Sigma \rightarrow Q$ gibt es genau einen $\delta_{det, A} = \delta$.

3.4 akzeptierte Sprache

Sei A ein EA. Die Sprache $L(A) := \{w \in \Sigma^* : \delta_A^*(s, w) \cap F \neq \emptyset\}$ ist die akzeptierte Sprache von A.

3.5 Regulär

Eine Sprache L heißt regulär, wenn es einen EA A mit $L(A)=L$ gibt.
Wir schreiben REG für die Klasse der regulären Sprachen.

Beispiel

Zu jedem Zeitpunkt während der Verbindung der Eingabe durch einen endlichen Automaten hängt der restliche Bearbeitung immer nur vom gegenwärtigen Zustand und dem noch einzulesenden Teil der Eingabe ab, nicht aber wie bei TM im allgemeinen von vergangenen Bandmanipulation. Interpretiert man die Eingabe als von einer äußeren Quelle kommend, so ist der Zustand des Automaten also allein durch seinen Zustand gegeben und der nächste Zustand hängt nur vom Zugeführten Symbol ab. Daher bietet sich eine Darstellung eines EA durch ein Übergangsdiagramm oder eine sogenannte Übergangstabelle an.

Sei $A := (\{q_0, q_1\}, \{0, 1\}, \Delta, q_0, \{q_1\})$ mit $\Delta = \{(q_0)\}$ Übergangsdiagramm und übergangstabelle von sehen wie folgt aus:

Zustand/Symbol	0	1
q_0	q_0	q_1
$q_1, *$	q_1	q_0

3.6 Übergangsdiagramm

Für jeden Zustand gibt es einen Kreis. Zustände in F bekommen einen Doppelkreis. Für $(q, a, q') \in \Delta$ für wir einen Pfeil von dem Kreis von q zu dem Kreis von q' mit der Beschriftung a.

Zustätzlich gibt es einen Pfeil (ohne Beschriftung) aus dem 'Nichts' zu dem Kreis des Startzustandes.

3.7 Potenzautomat

Sei A ein EA. Der Potenzautomat von A ist der DEA $P_a = (2^Q, \Sigma, \Delta', \{s\}, \{P \subseteq Q : P \cap F \neq \emptyset\})$ mit

$$\delta_{det, P_a}(Q_0, a) = \bigcup_{q \in Q_0} \delta_A(q, a) \quad \forall Q_0 \subseteq Q \quad \forall a \in \Sigma$$

Satz

Eine Sprache L ist genau dann regulär, wenn es einen DEA A mit $L(A) = L$ gibt.

Beweis. Sei $A = (Q, \Sigma, \Delta, s, F)$ ein EA mit Potenzautomat P_A . Es genügt zu zeigen, dass $L(A) = L(P_A)$. Hierfür genügt es zu zeigen, dass:

$$\delta_{det, P}^* = \delta_A^*(s, w) \quad \forall w \in \Sigma^*(*)$$

Denn damit folgt

$$\begin{aligned}
w \in L(P_A) &\Leftrightarrow \delta_{P_A}^*(\{s\}, w) \cap \{P \subseteq Q : P \cap F \neq \emptyset\} \neq \emptyset \\
&\Leftrightarrow \delta_{det, P_A}^*(\{s\}, w) \cap F \neq \emptyset \\
&\stackrel{(*)}{\Leftrightarrow} \delta_A^*(\{s\}, w) \cap F \neq \emptyset \\
&\Leftrightarrow w \in L(A)
\end{aligned}$$

Wir zeigen (*) mittels vollständiger Induktion über $|w|$. Es gilt $\delta_{det, P_A}^*(\{s\}, \lambda) = \delta_A^*(s, \lambda)$. Sei $w \in \Sigma^+$ mit $\delta_{det, A}^*(\{s\}, v) = \delta_A^*(s, v) \ \forall v \in \Sigma^{\leq |w|-1}$. Nun zeigen wir (*). Sei $va := w$ mit $a \in \Sigma$ und $|v| = |w| - 1$.

$$\begin{aligned}
\delta_{det, P_A}^*(\{s\}, w) &\stackrel{\text{Bem 4.5}}{=} \delta_{det, P_A}^*(\delta_{det, P_A}^*(\{s\}, v), a) \\
&\stackrel{\text{Ind. hyp}}{=} \delta_{det, P_A}^*(\delta_{det, P_A}^*(\{s\}, v), a) \\
&= \bigcup_{q \in \delta_{det, A}^*} \delta_A(q, a) \\
&= \delta_A^*(\delta_A^*(s, v), a) \\
&= \delta_A^*(s, va) \\
&= \delta_A^*(s, w)
\end{aligned}$$

□

4 Reguläre Sprachen

Wir verschieben den Fokus von endlichen Automaten auf die Klasse der von diesen erkannten Sprachen. Dabei spielen endl. Automaten weiterhin eine wichtige Rolle. Wegen 4.7 beschränken wir uns auf deterministische endliche Automaten.

Sei A DEA. Die Menge der zulässigen Eingaben Σ^ ist unendlich groß, die Menge der Zustände Q ist aber endlich. Zwangsläufig wird A also das Einlesen verschiedener Eingaben im gleichen Zustand abschließen (und somit gleich behandeln). Dies führt zum Begriff der A -Äquivalenz.*

4.1 Äquivalenzrelation

Sei A eine Menge. Eine Äquivalenzrelation auf A ist eine Relation $\sim \subseteq A \times A$, so dass die folgenden Eigenschaften erfüllt sind.

1. $a \sim a \quad \forall a \in A$ (Reflexivität)
2. $a \sim b \Rightarrow b \sim a \quad \forall a, b \in A$ (Symmetrie)
3. $a \sim b, b \sim c \Rightarrow a \sim c \quad \forall a, b, c \in A$ (Transitivität)

Die Äquivalenzklasse eines Elements $a \in A$ bezüglich \sim ist die Menge $[a]_\sim := \{a' \in A : a' \sim a\}$. Der Index von \sim ist die Kardinalität der Menge $A/\sim := \{[a]_\sim : a \in A\}$ falls diese endlich ist und unendlich andernfalls.

4.2 A-Äquivalenz

Sei A ein DEA. mit erweiterter Übergangsfunktion $\delta^* : Q \times \Sigma \rightarrow Q$. Die A-Äquivalenz ist die Relation \sim_A auf Σ^* mit

$$u \sim_A v \Leftrightarrow \delta^*(s, u) = \delta^*(s, v) \quad \forall u, v \in \Sigma^*$$

- (i) Die A-Äquivalenz ist eine Äquivalenzrelation.
- (ii) Der Index von \sim_A ist höchstens $|Q|$.
- (iii) Es gilt $L(A) = \bigcup_{w \in L(A)} [w]_{\sim_A}$.

4.3 Rechtskongruenz

Sei Σ ein Alphabet. Eine Rechtskongruenz auf Σ^* ist eine Äquivalenzrelation $\sim \subseteq (\Sigma^*)^2$ mit $u \sim v \Rightarrow uw \sim vw \quad \forall u, v, w \in \Sigma^*$.

Proposition

Sei $A = (Q, \Sigma, \Delta, s, F)$ ein DEA. Die A-Äquivalenz \sim_A ist eine Rechtskongruenz auf Σ^* .

Beweis. Seien $u, v, w \in \Sigma^*$ mit $u \sim_A v$. Dann gilt

$$\begin{aligned} \delta_{det,A}^*(s, uw) &= \delta_{det,A}^*(\delta_{det,A}^*(s, u), w) = \delta_{det,A}^*(\delta_{det,A}^*(s, v), w) \\ &= \delta_{det,A}^*(s, vw). \end{aligned}$$

□

Dann gilt $uw \sim_A vw$. □ Zu jedem DEA A gibt es also eine dazugehörige Rechtskongruenz \sim auf Σ^* mit endlichem Index so dass $L(A)$ die Vereinigung von Äquivalenzklasse von \sim_A ist. Tatsächlich gilt auch die Umkehrung: Ist L die Vereinigung von Äquivalenzklasse einer Rechtskongruenz \sim mit endlichem Index, so gibt es einen DEA A mit $L(A) = L$.

4.4

Sei Σ ein Alphabet und L Vereinigung von Äquivalenzklassen einer Rechtskongruenz \sim auf Σ^* mit endlichem Index. Es bezeichne

$$A_{\sim,L} := (\Sigma^*/\sim, \Sigma, \Delta, [\lambda]_{\sim}, \{[w]_{\sim} : w \in L\})$$

den DEA mit $\delta_{det, A_{\sim,L}}([w]_{\sim}, a) = [wa]_{\sim} \forall w \in \Sigma^*$ und $a \in \Sigma$.

Lemma

Sei Σ ein Alphabet, L Vereinigung von Äquivalenzklassen einer Rechtskongruenz \sim auf Σ^* mit endlichem Index und sei $\delta^* : \Sigma^*_{/\sim} \times \Sigma^* \rightarrow \Sigma^*_{/\sim}$ die erweiterte Übergangsfunktion von $A_{\sim,L}$. Dann gilt $\delta^*([\lambda]_{\sim}, w) = [w]_{\sim} \forall w \in \Sigma^*$.

Beweis. Wir verwenden vollständige Induktion über $|w|$. Es gilt $\delta^*([\lambda]_{\sim}, \lambda) = [\lambda]_{\sim}$. Sei nun $w \in \Sigma^+$... \square

Satz

Sei L die Vereinigung von Äquivalenzklasse einer Rechtskongruenz \sim mit endlichem Index. Es gibt $L(A_{\sim,L}) = L$.

Beweis. Sei Σ das Alphabet, so dass \sim eine Rechtskongruenz auf Σ^* ist. Sei $\delta^* : \Sigma^*_{/\sim} \times \Sigma^* \rightarrow \Sigma^*_{/\sim}$ die erweiterte Übergangsfunktion von $A_{\sim,L}$ und sei $w \in \Sigma^*$. Es folgt

$$\begin{aligned} w \in L(A_{\sim,L}) &\Leftrightarrow \delta^*([\lambda]_{\sim}, w) \in [v]_{\sim} : v \in L \\ &\Leftrightarrow [w]_{\sim} \in [v]_{\sim} : v \in L \\ &\Leftrightarrow \exists v \in L : [w]_{\sim} = [v]_{\sim} \\ &\Leftrightarrow \exists v \in L : w \sim v \\ &\Leftrightarrow w \in L \end{aligned}$$

...

\square

Korollar

Eine Sprache L ist genau dann regulär, wenn sie die Vereinigung von Äquivalenzklassen einer Rechtskongruenz mit endlichem Index ist.

Beweis. Folgt aus Bemerkung 5.3, Proposition 5.5 und Satz 5.8 \square

Betrachten man nur deterministische endliche Automaten ohne unerreichbare Zustände, so entsprechen diese bis auf Unbenutzung von Zuständen sogar den Rechtskongruenzen mit endlichem Index zusammen mit Vereinigung von Äquivalenzklassen dieser.

4.5 Erreichbar

Sei Σ ein Alphabet. Sei A ein EA mit erw. Übergangsfunktion δ^* . Ein Zustand $q \in Q$ heißt erreichbar in A wenn es ein Wort $w \in \Sigma^*$ mit $q \in \delta^*(s, w)$ gibt.

4.6 Isomorph

Sei A_i für $i \in \{1, 2\}$ ein EA mit Übergangsfunktion δ_i .

Die endlichen Automaten A_1 und A_2 sind Isomorph, kurz $A \cong A_2$, wenn es eine Projektion $f : Q_1 \rightarrow Q_2$ gibt, so dass folgendes gilt:

1. $f(s_1) = s_2$
2. $\delta_2(f(q_1), a) = f(\delta_1(q_1, a)) \forall q_1 \in Q, a \in \Sigma$
3. $f(F_1) = F_2$

Satz

- (i) Ist A eine DEA ohne un erreichbare Zustände, so gilt $A \cong A_{\sim A, L(A)}$
- (ii) Ist L die Vereinigung von Äquivalenzklasse einer Rechtskongruenz \sim mit endlichem Index, so gilt $(\sim, L) = (\sim_{A_{\sim, L, L(A_{\sim, L})}})$.

Beweis. (i) Sei $A = (Q, \Sigma, \Delta, s, F)$ eine DEA mit erweiterte Übergangsfunktion $\delta^* : Q \times \Sigma^* \rightarrow Q$ ohne un erreichbare Zustände, $\sim := \sim_A$, $A' := A_{\sim, L(A)}$ und sei $\delta' : \Sigma^* / \sim \times \Sigma^* \rightarrow \Sigma / \sim$ die erweiterte Übergangsfunktion von A' . Sei $f : Q \rightarrow \Sigma^* / \sim$ die Bijektive mit $f(q) := \{w \in \Sigma^* : \delta^*(s, w) = q\}$. Es gelte $f(s) = [\lambda]_{\sim}$ und $f(F) = \{[w]_{\sim} : w \in L(A)\}$. Es genügt somit zu zeigen, dass $\delta'(f(q), a) = f(\delta(q, a)) \forall q \in Q, a \in \Sigma$. Sei $q \in Q, a \in \Sigma^*$. Es genügt $w \in \delta'(f(q), a) \Leftrightarrow \delta^*(s, w) = \delta^*(q, a)$ zu zeigen. Sei $v \in \Sigma^*$ mit $\delta^*(s, v) = q$. Nun gilt $w \in \delta'(f(q), a) \Leftrightarrow w \in \delta'([v]_{\sim}, a) \Leftrightarrow w \sim va \Leftrightarrow \delta^*(s, w) = \delta^*(s, va) \Leftrightarrow \delta^*(s, w) = \delta^*(q, a)$

- (ii) Sei Σ ein Alphabet, \sim eine Rechtskongruenz auf Σ^* , L Vereinigung von Äquivalenzklassen von \sim , $A' := A_{\sim, L} = (\Sigma^* / \sim, \Sigma, A', [\lambda]_{\sim}, \uparrow)$, $\delta'^* : \Sigma^* / \sim \times \Sigma^* \rightarrow \Sigma^* / \sim$ die erweiterte Übergangsfunktion von A' $\{w \in \Sigma^* : w \in L\}$ und $\sim' := \sim_{A'}$. Nach Satz 5.8 gilt $L = L(A')$, es genügt also $\sim = \sim'$ zu zeigen. Sei $u, v \in \Sigma^*$. Aus lemma 5.7 folgt $u \sim v \Leftrightarrow [u]_{\sim} = [v]_{\sim} \Leftrightarrow \delta'(\dots) \dots$

□

4.7 L-Äquivalenz

Sei L eine Sprache über einem Alphabet Σ . Die L-Äquivalenz von L als Sprache ist die Relation \sim_L auf Σ^* mit

$$u \sim_L v \Leftrightarrow (uw \in L \Leftrightarrow vw \in L \forall w \in \Sigma^*)$$

Bemerkung

Sei L eine Sprache über Σ .

1. Die l-Äquivalenz ist eine Rechtskongruenz
2. Es gilt $L = \bigcup_{w \in L} [w]_{\sim_L}$

4.8 Partition

Sei A eine Menge. Eine Partition von A ist eine Menge $\mathcal{A} = \{A_1, \dots, A_n\}$ paarweise disjunkte nichtgeliche Teilmengen von A mit $\bigcup_{i \in [n]} A_i = A$

4.9 Verfeinerung

Seien \mathcal{A}_1 und \mathcal{A}_2 Partitionen einer Menge A . Die Partition \mathcal{A}_2 verfeinert \mathcal{A}_1 , wenn es $\forall A_2 \in \mathcal{A}_2$ ein $A_i \in \mathcal{A}_1$, mit.... Seien \mathcal{A}_1 und \mathcal{A}_2 Partitionen einer Menge A , so dass \mathcal{A}_2 die Partition \mathcal{A}_1 verfeinert.

1. $\forall A' \in \mathcal{A}_1$, gibt es eine Teilmenge $\mathcal{A}'_2 \in \mathcal{A}_2$, die eine Partition von A' ist
2. ...

Proposition

Seien Σ ein Alphabet und L eine Sprache über Σ und \sim eine Rechtskongruenz auf Σ^* mit $L = \bigcup_{w \in L} [w]_{\sim}$. Die Partition $\Sigma^* \setminus \sim$ ist eine Verfeinerung der Partition $\Sigma^* \setminus \sim_L$.

Beweis. Seien $u, v \in \Sigma^*$ mit $u \sim v$. Es genügt zu zeigen, dass $u \sim_L v$.

Sei $w \in \Sigma^*$. Es genügt $uw \in L \Leftrightarrow vw \in L$ zu zeigen.

Da \sim eine Rechtskongruenz ist folgt $uw \sim vw$. Ist $uw \in L$, so folgt aus $L = \bigcup_{w' \in L} [w']_{\sim}$ auch $vw \in L$ (analog folgt auch $vw \in L \Rightarrow uw \in L$).

$\Rightarrow u \sim_L v$

□

D.h. \sim_L ist die grösste Partition, die L darstellen kann.

4.10 Minimalautomat

Sei L eine reguläre Sprache über Σ . Der Minimalautomat von L als Sprache über Σ ist der DEA $A_{\sim_L, L}$

Satz

Sei L eine reguläre Sprache über Σ und sei M der Minimalautomat von L . Dann gilt:

1. $L(M) = L$
2. ist A ein DEA mit Zustandsmenge Q_A und $L(A) = L$, so gilt $|Q_A| \geq |Q|$
3. ist A ein DEA mit $|Q|$ Zuständen und $L(A) = L$, so gilt $A \cong M$