

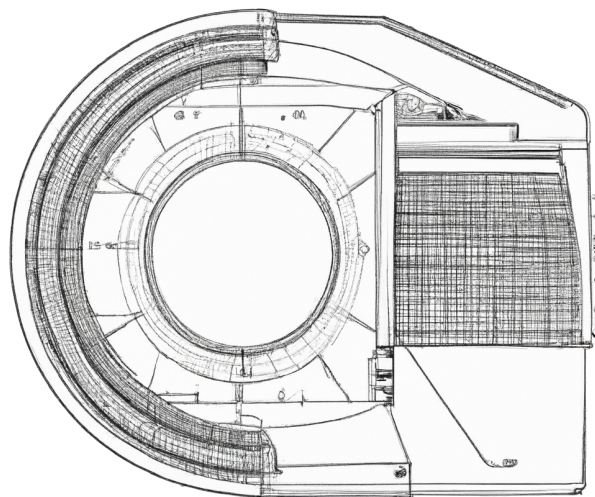
# Einführung in die Theoretische Informatik

Felix Ichtters, Lukas Dzielski\*

Sommersemester 2023

---

*Begleitmaterial mit den wichtigsten Definitionen und Aufgabenstellungen zur  
Vorlesung 'Einführung in die Theoretische Informatik'.*



---

\*Universität Heidelberg

# Inhaltsverzeichnis

1. Notationen	4
I. Definitionen	5
2. Grundlagen	5
2.1. Alphabet	5
2.2. Wörter	5
2.3. Sprache	5
2.4. $\Sigma^*$	5
2.5. Verkettung	5
2.6. Präfix, Infix, Suffix	5
2.7. Homomorphismus	6
2.8. Längenlexikographische Ordnungen	6
2.9. $\text{bin}(i)$	6
3. Turingmaschinen	7
3.1. Konfiguration	7
3.1.1. Nachfolgekonfiguration	7
3.2. Rechnung	8
3.3. Total	8
3.4. Akzeptierte Sprache	8
3.5. Entscheidbar	8
3.6. Rekursiv aufzählbar	8
3.7. Ausgabe	8
3.8. Berechnete Funktion	8
3.9. Partiell Berechnbar	9
3.10. Charakteristische Funktion	9
3.11. Normiert	9
3.12. Code	10
3.13. Standardaufzählung	10
3.14. $\mathcal{U}$	10
3.15. Universell	10
3.16. $s_n^m$ – Theorem	10
3.17. Diagonales Halteproblem	11
3.18. m-Reduktion	11
3.19. Postsches Korrespondenzproblem	11
3.20. Fixpunkt	11
3.21. Rekursionstheorem	11
3.22. Indexmenge	11
4. Automaten	12
4.1. Endlicher Automat	12
4.2. Übergangsfunktion eines EA	12
4.3. Übergangsfunktion eines DEA	12
4.4. akzeptierte Sprache	12
4.5. Regulär	12
4.6. Übergangsdiagramm	12

4.7. Potenzautomat . . . . .	13
<b>5. Reguläre Sprachen</b>	<b>13</b>
5.1. Äquivalenzrelation . . . . .	13
5.2. A-Äquivalenz . . . . .	13
5.3. Rechtskongruenz . . . . .	13
5.5. Erreichbar . . . . .	14
5.6. Isomorph . . . . .	14
 <b>II. Aufgabenstellungen</b>	 <b>15</b>

# 1. 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  $\varphi$  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

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

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

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

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

# Teil I.

## Definitionen

---

### 2. Grundlagen

#### 2.1. Alphabet

Ein Alphabet ist eine nichtleere Menge  $\Sigma$ , die Elemente heißen Symbole.

#### 2.2. Wörter

Ein Wort über einem Alphabet  $\Sigma$  ist eine endliche Folge von Symbolen aus  $\Sigma$ .

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  $w(i) = a_i \forall i \in [n]$ .

#### 2.3. Sprache

Eine Sprache ist eine Menge von Wörtern über einem gemeinsamen Alphabet  $\Sigma$ .

#### 2.4. $\Sigma^*$

Die Menge aller Wörter über  $\Sigma$  wird mit  $\Sigma^*$  bezeichnet.

Für  $n \in \mathbb{N}_0$  setzen wir

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

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

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

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

#### 2.5. 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

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

#### 2.6. 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  $v = wu$

## 2.7. Homomorphismus

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

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

## 2.8. 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  $u(i) \leq v(i)$

## 2.9. bin(i)

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

### 3. Turingmaschinen

Eine k-Band Turingmaschine ist ein Tupel

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

- $Q$  ist die endliche Zustandsmenge
- $\Sigma$  ist das Eingabealphabet
- $\Gamma$  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  $\Delta$  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

#### 3.1. 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

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

Die Stoppkonfiguration ist die Konfiguration zu der es keine Nachfolgekongfiguration gibt

##### 3.1.1. 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.

### 3.2. 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 Nachfolgekonfiguration 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  $start_M(w_1, \dots, w_n) = C_1, C_2, \dots, C_m$ .

### 3.3. Total

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

### 3.4. Akzeptierte Sprache

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

Die akzeptierte Sprache  $L(M)$  ist die Sprache über dem Alphabet  $\Sigma$ , 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*

### 3.5. Entscheidbar

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

### 3.6. Rekursiv aufzählbar

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

### 3.7. 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.*

### 3.8. Berechnete Funktion

Sei  $M$  eine k-DTM. Die von  $M$  berechnete  $n$ -äre partielle Funktion  $\varphi_M$  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.*

### 3.9. Partiiell Berechnbar

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

Ist  $\varphi$  total und partiell berechenbar, so ist  $\varphi$  berechenbar.

### 3.10. Charakteristische Funktion

Sei  $L$  eine Sprache über dem Alphabet  $\Sigma$ .

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

### 3.11. Normiert

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.

### 3.12. 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

$$code(\Delta) = code_{e_1}(\Delta) \dots code_{|\Delta|}(\Delta)$$

- 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.*

### 3.13. 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  $\varphi$ .

### 3.14. $\mathcal{U}$

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.

### 3.15. Universell

Eine DTM  $\mathcal{U}$  heißt universell, wenn es für alle  $n \in \mathbb{N}$  und alle partiell berechenbaren Funktionen  $\varphi : \mathbb{N}_0^n \rightsquigarrow \mathbb{N}_0$  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.

### 3.16. $s_n^m$ – 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  $e, x_1, \dots, x_m, y_1, \dots, y_n \in \mathbb{N}_0$ .

### 3.17. 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.*

### 3.18. m-Reduktion

Für eine Sprache  $A$  über einem Alphabet  $\Sigma$  und eine Sprache  $B$  über einem Alphabet  $\Gamma$  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  $A =_m B$ .*

### 3.19. Postsches Korrespondenzproblem

Für ein Alphabet  $\Sigma$  sei eine Instanz des Postschen Korrespondenzproblem über  $\Sigma$  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  $\Sigma$ , kurz  $PCP_\Sigma$ , ist die Menge aller lösbarer Instanzen des Postschen Korrespondenzproblems über  $\Sigma$ .

Für ein Alphabet  $\Sigma$  sei eine Instanz des modifizierten Postschen Korrespondenzproblems über  $\Sigma$  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  $\Sigma$ , kurz  $MPCP_\Sigma$  ist die Menge aller lösbarer Instanzen des modifizierten Postschen Korrespondenzproblems über  $\Sigma$ .

### 3.20. Fixpunkt

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

### 3.21. Rekursionstheorem

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

### 3.22. 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.

## 4. Automaten

### 4.1. Endlicher Automat

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

- $Q$  ist eine endliche Menge, die Zustandsmenge
- $\Sigma$  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  $(q, a, q')$
- $s \in Q$  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  $M_A = (Q, \Sigma, \Sigma \cup \{\square\}, \{(q, a, q', a, R) : (q, a, q') \in \Delta\}, s, F)$ .

### 4.2. 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  $\bigcup_{q \in Q_0} \delta_A^*(q, w)$ .

### 4.3. 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  $\delta_A^*$ .

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

### 4.4. 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  $A$ .

### 4.5. 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.

### 4.6. 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.

## 4.7. 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

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

## 5. 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  $\Sigma^*$  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.*

### 5.1. Äquivalenzrelation

Sei  $A$  eine Menge. Eine Äquivalenzrelation auf  $A$  ist eine Relation  $\sim \subseteq A$ , 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.

### 5.2. A-Äquivalenz

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

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

### 5.3. Rechtskongruenz

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

### 5.4.

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

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

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

## 5.5. Erreichbar

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

## 5.6. Isomorph

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

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.  $\delta_2(f(q_1), a) = f(\delta_1(q_1, a)) \quad \forall q_1 \in Q_1$
3.  $f(F_1) = F_2$

## Teil II.

# Aufgabenstellungen

---