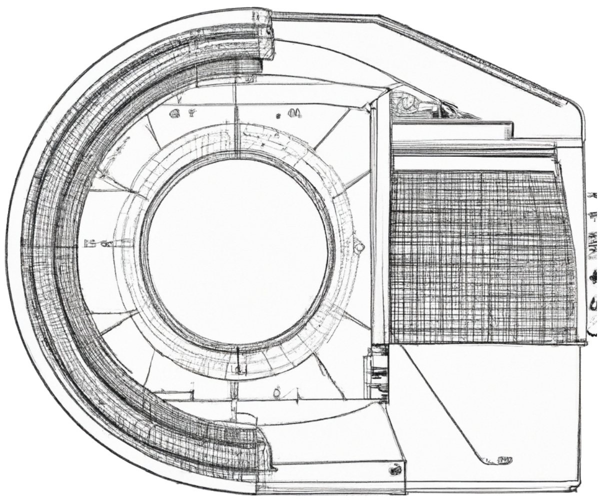
Einführung in die Theoretische Informatik

Felix Ichters, Lukas Dzielski*

Sommersemester 2023

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



*Universität Heidelberg

Inhaltsverzeichnis

1. Notationen

- Für $n \in \mathbb{N}_0$, sei $[n] = \{1, \ldots, n\}$ und $[n]_0 = \{0, 1, \ldots, n\}$.
- Für eine Menge A und $n \in \mathbb{N}$ ist $A^n = \{(a_1, \ldots, a_n) : a_1, \ldots, a_n \in A\}$.
- Für $n \in \mathbb{N}$ ist eine n-äre partielle Funktion $\varphi : A^n \rightsquigarrow B$ eine Funktion mit $dom(\varphi) \subseteq A^n$ und $Im(\varphi) \subseteq B$. Für $a_1, \ldots, a_n \in A$ bedeutet $\varphi(a_1, \ldots, a_n) \downarrow$, dass $a_1, \ldots, a_n \in dom(\varphi)$ gilt und $\varphi(a_1, \ldots, a_n) \uparrow$ beduetet, dass $(a_1, \ldots, a_n) \notin dom(\varphi)$. Die partielle Funktion φ ist total, wenn $dom(\varphi) = A^n$ 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

  $a \leq a \ \forall a \in A$ (Reflexivität)

  $a \leq b \wedge b \leq a \Rightarrow a = b \ \forall a, b \in A$ (Antisymmetrie)

  $a \leq b, b \leq c \Rightarrow a \leq c \ \forall a, b, c \in A$ (Transitivität)

  $a \leq b \vee b \leq a \ \forall a, b \in A$ (Totalität)

Teil I.
Definitionen

2. Grundlagen

2.1. Alphabet

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

2.2. 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 $a_1, \ldots, a_n \in \Sigma$ bezeichnet $a_1, a_2, \ldots, a_n$ 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 Σ.

2.4. Σ^*

Die Menge aller Wörter über Σ wird mit Σ^* bezeichnet.
Für $n \in \mathbb{N}_0$ setzen wir

$$\Sigma^{\leq n} := \{w \in \Sigma^* : |w| \leq n\}$$
$$\Sigma^{=n} := \{w \in \Sigma^* : |w| = n\}$$
$$\Sigma^{\geq n} := \{w \in \Sigma^* : |w| \geq n\}$$
$$\Sigma^{+} := \Sigma^{\geq 1}$$

2.5. Verkettung

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

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

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 $\varphi : L \to M$ Homomorphismus von Sprachen, wenn gilt:

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

2.8. Längenlexikographische Ordnungen

Es gilt $u \leq_{llex} v$ 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 Längenlexikopraphischer Reihnfolge $i+1$-te Binärwort.
$1bin(i)$ beschreibt die Binärdarstellung von $i+1$.

3. Turingmaschienen

Eine k-Band Turingmaschiene 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 $\Box \in \Gamma \backslash \Sigma$
- $\Delta \subseteq Q \times \Gamma^k \times Q \times \Gamma^k \times \{L, S, R\}^k$ 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

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

$$\mathit{start}(u_1, \dots, u_n) = (s, u_1 \Box u_2 \Box \dots \Box u_n, \Box, \dots, \Box, 1, \dots, 1)$$

Die Stoppkonfiguartion ist die Konfiguration zu der es keine Nachfolgekonfiguration gibt

3.1.1. Nachfolgekonfiguration

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' **Nachfolgekonfiguration** 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

$$w_i = \begin{cases} \Box a'_i w_i(2) \dots w_i(|w_i|) & \text{falls } p_i = 1 \text{ und } B_i = L \\ w_i(1) \dots w_i(|w_i| - 1) a'_i \Box & \text{falls } p_i = |w_i| \text{ und } B_i = R \\ w_i(1) \dots w_i(p_1 - 1) a'_i w_i(p_i + 1) \dots w_i(|w_i|) & \text{sonst} \end{cases}$$

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

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 Σ, so dass $\forall w \in \Sigma^*$ genau dann $w \in L(M)$ gilt, wenn es eine endliche Rechnung $C_1, \ldots, C_n$ zur Eingabe w gibt, bei der C_n eine akzeptierte Stoppkonfiguartion ist.

Für nicht-deterministische Tms heißt das, dass es für die Wörter in der akzeptierten Sprache nur mind. eine in einer akzeptierten Stoppkonfiguartion 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 Stoppkonfiguartion 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 $w \sqsubseteq w_1(p_1) \ldots w_1(|w_1|)$ max. Länge mit $w \in (\Gamma \backslash \{\Box\})^*$.

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

3.8. Berechnete Funktion

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

$$\varphi_M : (\Sigma^*)^n \rightsquigarrow (\Gamma \backslash \{\Box\})^*$$

sodass folgendes gilt:

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

2. Ist die Rechnung zur Eingabe $w_1, \ldots, w_n$ unendlich, so gilt $\varphi_M(w_1, \ldots, w_n) \uparrow$

Nur deterministische TMs können Funktionen berechnen.

3.9. Partiell 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.

3.10. Charakteristische Funktion

Sei L eine Sprache über dem Alphabet Σ.

1. Die charakteristische Funktion von L als Sprache über Σ ist die Funktion $\mathbb{1}_L : \Sigma^* \to \{0, 1\}$ mit $\mathbb{1}_L(w) = 1\ \forall w \in L$ und $\mathbb{1}_L(w) = 0\ \forall w \in \Sigma^* \backslash L$.

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 $\chi_L(w) \uparrow \forall w \in \Sigma^* \backslash L$.

3.11. Normiet

Eine 1-DTM M heißt normiert, wenn Q=0,...,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

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

  $code(I) = 0^{|bin(q)|}1bin(q)a0^{|bin(q')|}1bin(q')a'code(B)$

- Codieren des Instruktinssatzes

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

- Codieren der normierten TM

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

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 längenlexikopraphischer Ordnung.*Für

$e \in \mathbb{N}$ sei $\mathcal{M}_e$ die durch $\hat{w}_e$ codierte TM und für $n \in \mathbb{N}$ sei $\Phi_e^n : \mathbb{N}_0^n \to \mathbb{N}_0$ 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 $\varphi : \mathbb{N}_0^n \to \mathbb{N}_0$ heißt jede Zahl $e \in \mathbb{N}_0$ mit $\Phi_e^n = \varphi$ Index von φ.

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 Asugabe der Simulation ausgibt.

3.15. Universell

Eine DTM $\mathcal{U}$ heßt 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 $mathcalU(e, x_1, \dots, x_n) = \varphi(x_1, \dots, x_n)$ 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 $s_n^m : \mathbb{N}_0^{m+1} \to \mathbb{N}_0$ 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 Σ und eine Sprache B über einem Alphabet Γ ist A genau dann many-one-reduzierbar, auch m-reduzierbar, auf B, kurs $A \leq_m B$, wenn es eine berechenbare Funktion $f : \Sigma^* \to \Gamma^*$ 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-äquibalent, kurz $A =_m B$.

3.19. 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 $(u_1, v_1), \ldots, (u_n, v_n)$ von Paaren in I mit $n \geq 1$, so dass

$$u_1 \ldots u_n = v_1 \ldots v_n$$

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 ein solche Instanz ist eine endliche Folge $(u_1, v_1), \ldots, (u_n, v_n)$ von Paaren in I, so dass

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

Gibt es ein 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.

3.20. Fixpunkt

Ein Fixpunkt einer berechenbaren Funktion $f : \mathbb{N}_0 \to \mathbb{N}_0$ 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
- 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')
- $s \in Q$ 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* $M_A = (Q, \Sigma, \Sigma \cup \{\Box\}, \{(q, a, q', a, R) : (q, a, q') \in \Delta\}, s, F)$.

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 $\delta_{det,A} : Q \times \Sigma \to Q$ mit $\delta_a(q, a) = \{\delta_{drt,A}(q, a)\}$ $\forall q \in Q$ 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 $\delta^*_{det,A}(Q_0, w)$.

4.4. akzeptierte Sprache

Sei A ein EA. Die Sprache $L(A) := \{w \in \Sigma^* : \delta^*_A(s, w) \cap F \neq \varnothing\}$ 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 $P_a = (2^Q, \Sigma, \Delta', \{s\}, \{P \subseteq Q : P \cap F \neq \varnothing\})$ mit

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

Teil II.
Aufgabenstellungen