Formale Sprachen und Automaten
(Formelsammlung, v1)

Uwe Nestmann
Technische Universität Berlin

29. März 2021

Literatur

[AB02] Alexander Asteroth and Christel Baier. *Theoretische Informatik*. Pearson Studium, München, 2002.

[EMC+01] H. Ehrig, B. Mahr, F. Cornelius, M. Grosse-Rhode, and P. Zeitz. *Mathematisch-strukturelle Grundlagen der Informatik, 2. überarbeitete Auflage*. Springer, 2001.

[Ros07] Kenneth H. Rosen. *Discrete Mathematics and Its Applications*. McGraw-Hill, 2007.

[Sch01] Carol Schumacher. *Chapter Zero — Fundamental Notions of Abstract Mathematics*. Addison Wesley, 2001.

[Sch03] Uwe Schöning. *Theoretische Informatik — kurzgefasst*. Spektrum Lehrbuch. Spektrum Akademischer Verlag, 2003. 4. Auflage, korrigierter Nachdruck.

[SL19] Olivier Sète and Jörg Liesen. *Analysis I und Lineare Algebra für Ingenieurwissenschaften*. Technische Universität Berlin, 2019.

[Vel06] Daniel J. Velleman. *How to Prove It — A Structured Approach*. Cambridge University Press, 2006.

Inhaltsverzeichnis

0 Basiswissen

Die Inhalte dieses Kapitels sind vor allem als Nachschlagewerk zu verstehen: wenn sich Lesende der späteren Kapitel der Definition des einen oder anderen mathematischen Konzepts nicht sicher sind, so findet sich hier eine meist recht formale saubere Definition.

Der Text beinhaltet auch Passagen, die in einem leichten Grauton anstelle von Schwarz erscheinen. Dies deutet darauf hin, dass es sich um nützliche Einsichten oder Notationen handelt, die aber nicht wesentlich zum Verstehen der Inhalte des Moduls sind.

0.1 Mathematische Grundlagen Ana1/LinA

Wir fassen kurz zusammen, welche Inhalte in Bezug auf das Skript des Kombimoduls „Analysis I und Lineare Algebra für Ingenieurwissenschaften“ [SL19] schon bekannt sein sollten. Wir zeigen zudem auf, wo sich diese Teile in der vorliegenden Formelsammlung wiederfinden und worin ein meist höherer Grad der Formalisierung liegt.

- Dir Grundlagen der Mengenlehre sind aus dem Kombimodul bekannt. In Abschnitt 0.3 fassen wir diese formal zusammen und betonen die verschiedenen Methoden, mit denen Mengen definiert werden können. Dazu führen wir das in der Informatik eminent wichtige Konzept der Potenzmenge ein. Der Zusatzabschnitt 0.8 erweitert diese Möglichkeiten mithilfe von Abbildungen.

- Wir gehen davon aus, dass die üblichen Zahlenräume bzw. die entsprechenden Mengen bekannt sind (ebenfalls Abschnitt 0.3).

- Der Abschnitt 0.4 zur Aussagenlogik ist deutlich formaler gehalten als es aus dem Kombimodul bekannt ist; zudem wird auch deren Semantik formalisiert. Neben nützlicher Terminologie werden auch logische Äquivalenzen eingeführt, die die Grundlage und Berechtigung für viele bekannte Beweisprinzipien der Mathematik darstellen.

- Der Abschnitt 0.5 zur Prädikatenlogik ist ebenfalls recht formal gehalten, verzichtet aber der Einfachheit halber auf eine formale Semantik. Es genügt jedoch, um Induktionsprinzipien wie die aus der Mathematik bekannte vollständige Induktion als geschlossene Formel darzustellen.

- Der Abschnitt 0.7 zu Abbildungen und deren Eigenschaften orientiert sich weitgehend an der Notation des Kombimoduls. Abbildungen werden in diesem Modul jedoch als Spezialisierung des Relationskonzepts aus Abschnitt 0.6 eingeführt und vor allem im Abschnitt 0.9 zur Formalisierung von Kardinalitäten genutzt.

Insbesondere die Abschnitte 0.6 und 0.10 zu Relationen, darauf aufbauend zu Ordnungen 0.11 und Äquivalenzen 0.12, aber auch zu Kardinalitäten 0.9 enthalten im Vergleich zum Kombimodul neues Material.

0.2 Gleichheit[en]

In vielen Texten findet man „:=“ (gelegentlich auch „=:“) oder „$\stackrel{\text{def}}{=}$“ als Symbol für *definierte* bzw. *definierende Gleichheit*. In dieser Formelsammlung benutzen wir stattdessen das Symbol „$\triangleq$“. Beispielsweise wird durch

$$\text{Bez} \triangleq \textit{Ausdruck}$$

definiert, dass in der Folge – zumindest im Kontext dieser Definition – der Bezeichner Bez metasprachlich immer durch den Ausdruck *Ausdruck* ersetzt werden darf. Dies gilt dann auch nur, weil wir es für unsere Bedürfnisse so festgelegt, eben definiert haben.

Wir verwenden das eher generische Gleichheitssymbol „=“ immer dann, wenn aus irgendwelchen Gründen – neben der direkt definierten Gleichheit beispielsweise aufgrund der in der Schule gelernten Grundrechenarten – eine „wie auch immer geartete Gleichheit“ zwischen zwei Dingen bzw. Ausdrücken *festgestellt werden kann* und damit *gilt*.

Manche Gleichheiten bezeichnen wir auch als *Äquivalenz* (aka: *Gleichwertigkeit*) und verwenden dann dafür ein besonderes Symbol, oftmals das „$\equiv$“, wie beispielsweise im Fall der *semantischen Äquivalenz* von aussagen- und prädikatenlogischen Formeln in § 0.4 und 0.5. Was der Begriff abstrakt (und damit: genau) bedeutet, das zeigen wir in § 0.12.

Umgangssprachlich mag es helfen sich zu vergegenwärtigen, dass *das gleiche* und *dasselbe* unterschiedlich benutzt werden, obwohl beide den Charakter einer Gleichheit besitzen.

Etwas formaler: Nehmen wir beispielsweise die Menge der Brüche, wie in „Ana1/LinA“ [SL19] definiert, dann sind Brüche wie $\frac{2}{3}$ und $\frac{4}{6}$ definitiv verschiedene Gegenstände, also *nicht gleich*, sie bezeichnen aber die gleiche (sogar *dieselbe!*) rationale Zahl, die man ansatzweise mit der Periodendarstellung $0,\overline{6}$ eindeutig zu repräsentieren versucht. (Hier sei die Problematik der beiden verschiedenen Dezimaldarstellungen $0,\overline{9}$ und $1,0$ genannt, die ja dieselbe Zahl bedeuten.) Andererseits kann man $\frac{2}{3}$ und $\frac{4}{6}$ insofern auch als *äquivalent* in Bezug auf Kürzbarkeit ansehen.

0.3 Mengen

Die Bezeichner von Standardmengen, wie beispielsweise $\mathbb{N}$ für die Menge der natürlichen Zahlen, verwenden einen anderen Schriftsatz als Metavariablen für Mengen; für letztere bevorzugen wir $A, B, C, \ldots$, möglicherweise mit Indizes (Zahlen oder anderen Buchstaben) versehen.

0.3.1 Notation (Beschreibungsformen 1)

- Eine Menge A kann durch $A \triangleq \{ a_1,\ a_2,\ a_3,\ \ldots \}$ als die explizite Aufzählung unterscheidbarer Elemente a_i definiert werden.
- Die Aussage „$a \in A$“ bezeichnet den Sachverhalt, dass a ein Element der Menge A ist.
- Die *leere Menge* $\emptyset \triangleq \{\}$ enthält keine Elemente.

0.3.2 Definition (Zahlenmengen)

- $\mathbb{N} \triangleq \{ 0,\ 1,\ 2,\ 3,\ \ldots \}$; $\mathbb{N}^+ \triangleq \{ 1,\ 2,\ 3,\ \ldots \}$.
- Mit $\mathbb{Z}, \mathbb{Q}, \mathbb{R}$ bezeichnen wir die gebräuchlichen Mengen von *ganzen Zahlen*, *rationalen Zahlen*, *reellen Zahlen*.

0.3.3 Definition (Größe von Mengen) Sei A eine beliebige Menge. $\#(A)$ (oder auch $|A|$) bezeichnet die Anzahl der Elemente in A. Es gilt $\#(A) \in \mathbb{N}$ oder $\#(A) = \infty$.

0.3.4 Notation (Beschreibungsformen 2) Die Schreibweise

$$\{ x \in B \mid P(x) \} \quad \text{bzw.} \quad \{ x \mid P(x) \}$$

definiert die Menge derjenigen Elemente x [aus der Menge B], die die Eigenschaft bzw. das „Prädikat“ P erfüllen. Dabei gilt:

- Die Angabe von B kann manchmal entfallen, wird aber empfohlen. Das Symbol „$|$“ kann dabei als „so dass“ gelesen werden.
- Die Angabe von $P(x)$ erfolgt entweder umgangssprachlich oder durch eine logische Formel (siehe 0.3.7 sowie § 0.4 und 0.5).
- Die Menge $\{ x_i \mid i \in I \}$ heißt *über* I *indizierte Menge*.

Sowohl $\{ x \in B \mid P(x) \}$ als auch $\{ x \mid P(x) \}$ definieren nur dann Mengen, wenn für alle betroffenen x die Wahrheit von $P(x)$ überprüfbar ist.

0.3.5 Proposition (Mengen und Elemente)

1. Für beliebige Prädikate P gilt: $\{ y \mid P(y) \} = \{ x \mid P(x) \}$.
2. Für beliebige Mengen A gilt: $A = \{ x \mid x \in A \}$.
3. Für beliebige Prädikate P gilt: $\{ y \mid P(y) \} = \{ x \mid x \in \{ y \mid P(y) \} \}$.

0.3.6 Definition (Zahlenmengen)
$[m, n] \triangleq \{ i \in \mathbb{N} \mid m \leqslant i \text{ und } i \leqslant n \}$ für $m, n \in \mathbb{N}$

0.3.7 Notation („Logik-Sprache“ durch Junktor-Symbole)
Wir führen logische Konnektive zunächst semi-formal
als symbolische Abkürzungen natürlichsprachlicher Begriffe ein.

- *Verum* $\top$ steht für „wahr“
- *Falsum* $\bot$ steht für „falsch“
- *Negation* $\neg$ steht für „nicht“
- *Konjunktion* $\wedge$ steht für „und“
- *Disjunktion* $\vee$ steht für „oder“
- *Implikation* $\to$ steht für „impliziert“ (d.h. „läßt schließen auf“)
- *Biimplikation* $\leftrightarrow$ steht für „genau dann, wenn“ („g.d.w.“)
- *Universelle Quantifikation* $\forall$ steht für „für alle“
- *Existentielle Quantifikation* $\exists$ steht für „es gibt“

0.3.8 Definition (Teilmengen, Gleichheit von Mengen)
Seien A und B zwei beliebige Mengen.
A heißt *Teilmenge* von B, geschrieben $A \subseteq B$, wenn für alle x gilt:

$$(x \in A) \to (x \in B)$$

A und B heißen *gleich*, geschrieben $A = B$, wenn gilt:

$$A \subseteq B \ \wedge \ B \subseteq A$$

A heißt *echte Teilmenge* von B, geschrieben $A \subset B$, wenn gilt:

$$A \subseteq B \ \wedge \ A \neq B$$

Wenn A (echte) Teilmenge von B ist,
dann heisst B *(echte) Obermenge* von A.

0.3.9 Definition (Vereinigung, Durchschnitt, Komplement, Produkt)
Seien A und B zwei beliebige Mengen. Wir definieren:
Vereinigung[smenge] von A und B:

$$A \cup B \triangleq \{ x \mid x \in A \vee x \in B \}$$

Schnitt[menge] von A und B:

$$A \cap B \triangleq \{ x \mid x \in A \wedge x \in B \}$$

Differenz bzw. *Komplement* von A *in [Bezug auf]* B:

$$B \setminus A \triangleq \{ x \mid x \in B \wedge x \notin A \}$$

[kartesisches] Produkt von A und B:

$$A \times B \triangleq \{ (a,\ b) \mid a \in A \wedge b \in B \}$$

disjunkte Vereinigung von A und B:

$$A \uplus B \triangleq (A \times \{1\}) \ \cup \ (B \times \{2\})$$

Die Wahl von $\{1,\ 2\}$ in der Definition disjunkter Vereinigung ist willkürlich. Beliebige andere Werte hätten verwendet werden können.

0.3.10 Definition Sei $\boldsymbol{M}$ eine Menge von Mengen. Dann gilt:

$$\begin{aligned} \bigcup \boldsymbol{M} &\triangleq \{\, x \mid \exists m \in \boldsymbol{M} \;.\; x \in m \,\} \\ \bigcap \boldsymbol{M} &\triangleq \{\, x \mid \forall m \in \boldsymbol{M} \;.\; x \in m \,\} \end{aligned}$$

Als Spezialfälle werden häufig indizierte Vereinigungen verwendet.

$$\begin{aligned} \textstyle\bigcup_{i\in I} m_i &\triangleq \{\, x \mid \exists i \in I \;.\; x \in m_i \,\} \\ \textstyle\bigcap_{i\in I} m_i &\triangleq \{\, x \mid \forall i \in I \;.\; x \in m_i \,\} \end{aligned}$$

0.3.11 Definition (Potenzmenge) Sei A eine beliebige Menge. Dann heißt

$$\mathcal{P}(A) \triangleq \{\, B \mid B \subseteq A \,\}$$

Potenzmenge von A. Alternativ gilt die Notation $2^A \triangleq \mathcal{P}(A)$.

0.3.12 Proposition Sei A eine beliebige Menge. Dann gilt:

$$\#(\mathcal{P}(A)) = \begin{cases} 2^{\#(A)} & \text{falls } \#(A) < \infty \\ \infty & \text{falls } \#(A) = \infty \end{cases}$$

0.3.13 Notation (Beschreibungsformen 3: (Strukturelle) Induktion)
Unendlich große Mengen können auch *rekursiv* bzw. *induktiv* durch die Angabe endlich vieler Regeln eindeutig definiert werden. So ergibt sich die Menge der natürlichen Zahlen $\mathbb{N}$ aus folgenden zwei „Regeln“:

Basis $0 \in \mathbb{N}$;
Komposition wenn $n \in \mathbb{N}$, dann auch $n+1 \in \mathbb{N}$.

Gemeint ist damit, dass die Menge $\mathbb{N}$ aus *genau* solchen Elementen besteht, die sich durch die beiden Regeln „herleiten“ lassen. Mit anderen Worten: *alle* solchen Elemente gehören dazu, aber keine anderen.

Verallgemeinert spricht man von der Definition einer Menge M durch *strukturelle Induktion*, wenn es eine Basismenge B gibt, deren Elemente ohne weitere Vorbedingung dazugehören:

Basis $B \subseteq M$

Hinzu kommen endlich viele Regeln der **Komposition**, mit deren Hilfe aus in M bereits enthaltenen Elementen strukturell – wie im Beispiel der Definition von $\mathbb{N}$ ersichtlich – neue Elemente konstruiert in in M aufgenommen werden. Definition 0.4.1 liefert hierzu ein typisches Beispiel.

0.4 Aussagenlogik

0.4.1 Definition (Syntax der Aussagenlogik)
Sei V eine Menge von *aussagenlogischen Variablen*.
Sei $V \cap \{ \top, \bot, \neg, \wedge, \vee, \rightarrow, \leftrightarrow, (,) \} = \emptyset$.
Dann ist die *Menge* $\mathbf{A}(V)$ *der aussagenlogischen Formeln zu* V
definiert als kleinste Menge mit:

- $V \cup \{ \top, \bot \} \subseteq \mathbf{A}(V)$;
- wenn $\varphi \in \mathbf{A}(V)$, dann auch $\neg\varphi \in \mathbf{A}(V)$;
- wenn $\{ \varphi_1, \varphi_2 \} \subseteq \mathbf{A}(V)$,
  dann auch $\{ (\varphi_1 \wedge \varphi_2), (\varphi_1 \vee \varphi_2), (\varphi_1 \rightarrow \varphi_2), (\varphi_1 \leftrightarrow \varphi_2) \} \subseteq \mathbf{A}(V)$.

Wir verwenden $p, q, \ldots$ als Metavariablen für Elemente von V.
Wir verwenden $\varphi, \psi, \ldots$ als Metavariablen für für Elemente von $\mathbf{A}(V)$.

Formeln in $\mathbf{A}(V)$, die

- keine aussagenlogischen Variablen beinhalten, heißen *Aussagen*.
- aussagenlogische Variablen beinhalten, heißen *Aussagenschemata*.
- nur aus einem einzigen Element in $V \cup \{ \top, \bot \}$ bestehen,
  heißen *atomar*, alle anderen Formeln heißen *zusammengesetzt*.
- maximal eine aussagenlogische Variable und außer $\top$ oder $\bot$
  maximal einen weiteren Operator betreffen, nennen wir *trivial*.

0.4.2 Notation Für bessere Lesbarkeit können manche Klammern entfallen.
Dazu gelten folgende Regeln des *Operatorenvorrangs*:

- $\neg$ bindet stärker als alle anderen Konnektive;
- $\wedge$ und $\vee$ binden gleich stark;[1]
- $\wedge$ und $\vee$ binden stärker als $\rightarrow$ und $\leftrightarrow$;
- $\rightarrow$ und $\leftrightarrow$ binden gleich stark.

Außenklammern dürfen (nur zwecks Lesbarkeit) weggelassen werden.
Syntaktisch korrekte Formeln beinhalten dennoch immer alle Klammern.

0.4.3 Definition (Semantik der Aussagenlogik)
Sei V eine Menge von aussagenlogischen Variablen.
Eine *Variablenbelegung* β weist jedem Element $p \in V$
genau ein Element $\beta(p) \in \mathbb{B} \triangleq \{1, 0\}$ der *Wahrheitswerte* zu.
Die zu β gehörige *Auswertung* $[\![\cdot]\!]^\beta$ weist jedem Element $\varphi \in \mathbf{A}(V)$
genau ein Element $[\![\varphi]\!]^\beta \in \mathbb{B}$ zu.($[\![\psi]\!]^\beta$ darf kurz als $[\![\psi]\!]$ geschrieben
werden, wenn keine Verwechslungsgefahr besteht.)

Zur Definition von $[\![\cdot]\!]^\beta$ gilt für atomare Formeln:

- $[\![\top]\!]^\beta \triangleq 1$
- $[\![\bot]\!]^\beta \triangleq 0$
- $[\![p]\!]^\beta \triangleq \beta(p)$

[1] In manchen Darstellungen wird $\wedge$ als vorrangig gegenüber $\vee$ betrachtet.

Wahrheitstabellen definieren die Semantik zusammengesetzter Formeln (wobei wir der Einfachheit halber hier das Superskript β weglassen):

$[\![\varphi]\!]$	$[\![\neg\varphi]\!]$
0	1
1	0

$[\![\varphi_1]\!]$	$[\![\varphi_2]\!]$	$[\![\varphi_1 \wedge \varphi_2]\!]$
0	0	0
0	1	0
1	0	0
1	1	1

$[\![\varphi_1]\!]$	$[\![\varphi_2]\!]$	$[\![\varphi_1 \vee \varphi_2]\!]$
0	0	0
0	1	1
1	0	1
1	1	1

$[\![\varphi_1]\!]$	$[\![\varphi_2]\!]$	$[\![\varphi_1 \to \varphi_2]\!]$
0	0	1
0	1	1
1	0	0
1	1	1

$[\![\varphi_1]\!]$	$[\![\varphi_2]\!]$	$[\![\varphi_1 \leftrightarrow \varphi_2]\!]$
0	0	1
0	1	0
1	0	0
1	1	1

0.4.4 Bemerkung Es gilt offensichtlich, dass:

- $[\![\neg\varphi]\!] = 1 - [\![\varphi]\!]$
- $[\![\varphi_1 \wedge \varphi_2]\!] = [\![\varphi_1]\!] \cdot [\![\varphi_2]\!]$
- $[\![\varphi_1 \vee \varphi_2]\!] = [\![\varphi_1]\!] + [\![\varphi_2]\!] - [\![\varphi_1 \wedge \varphi_2]\!]$
- $[\![\varphi_1 \to \varphi_2]\!] = 1$ genau dann, wenn $[\![\varphi_1]\!] \leqslant [\![\varphi_2]\!]$
- $[\![\varphi_1 \leftrightarrow \varphi_2]\!] = 1$ genau dann, wenn $[\![\varphi_1]\!] = [\![\varphi_2]\!]$

0.4.5 Definition
Sei V eine Menge von *aussagenlogischen Variablen*.

1. Zwei Formeln $\varphi, \psi \in \mathbf{A}(V)$
   heißen *logisch äquivalent*, geschrieben $\varphi \equiv \psi$,[2]
   wenn für alle Variablenbelegungen β gilt: $[\![\varphi]\!]^\beta = [\![\psi]\!]^\beta$.
2. Eine Formel φ heißt *allgemeingültig*, wenn $\varphi \equiv \top$.
3. Eine Formel φ heißt *kontradiktorisch*, wenn $\varphi \equiv \bot$.
4. Eine Formel φ heißt *erfüllbar*,
   wenn es eine Variablenbelegung β gibt mit $[\![\varphi]\!]^\beta = 1$.

Äquivalenzen zwischen trivialen Formeln dürfen ohne Beweis benutzt werden (z.B.: $\top \wedge p \equiv p$, $p \vee \bot \equiv p$, $p \to p \equiv \top$, ...).

0.4.6 Proposition (Logische Äquivalenz (I))

$\varphi_1 \wedge \varphi_1 \equiv \varphi_1$	Idempotenz von $\wedge$
$\varphi_1 \vee \varphi_1 \equiv \varphi_1$	Idempotenz von $\vee$
$\varphi_1 \wedge \varphi_2 \equiv \varphi_2 \wedge \varphi_1$	Kommutativität von $\wedge$
$\varphi_1 \vee \varphi_2 \equiv \varphi_2 \vee \varphi_1$	Kommutativität von $\vee$
$(\varphi_1 \wedge \varphi_2) \wedge \varphi_3 \equiv \varphi_1 \wedge (\varphi_2 \wedge \varphi_3)$	Assoziativität von $\wedge$
$(\varphi_1 \vee \varphi_2) \vee \varphi_3 \equiv \varphi_1 \vee (\varphi_2 \vee \varphi_3)$	Assoziativität von $\vee$
$(\varphi_1 \wedge \varphi_2) \vee \varphi_3 \equiv (\varphi_1 \vee \varphi_3) \wedge (\varphi_2 \vee \varphi_3)$	Distributivität von $\vee$ über $\wedge$
$(\varphi_1 \vee \varphi_2) \wedge \varphi_3 \equiv (\varphi_1 \wedge \varphi_3) \vee (\varphi_2 \wedge \varphi_3)$	Distributivität von $\wedge$ über $\vee$

[2] In manchen Darstellungen wird $\equiv$ als Synonym für $\leftrightarrow$ betrachtet, d.h., als Junktor innerhalb der Formelsprache.

$$\begin{array}{rl}
\neg\neg\varphi_1 \equiv \varphi_1 & \text{doppelte Negation} \\
\varphi_1 \wedge \neg\varphi_1 \equiv \bot & \text{Tertium Non Datur } \wedge \\
\varphi_1 \vee \neg\varphi_1 \equiv \top & \text{Tertium Non Datur } \vee \\
\neg(\varphi_1 \wedge \varphi_2) \equiv \neg\varphi_1 \vee \neg\varphi_2 & \text{De Morgan I} \\
\neg(\varphi_1 \vee \varphi_2) \equiv \neg\varphi_1 \wedge \neg\varphi_2 & \text{De Morgan II}
\end{array}$$

$$\begin{array}{rl}
\varphi_1 \to \varphi_2 \equiv \neg\varphi_1 \vee \varphi_2 & \text{Implikation} \\
\varphi_1 \to \varphi_2 \equiv \neg\varphi_2 \to \neg\varphi_1 & \text{Kontraposition} \\
\varphi_1 \leftrightarrow \varphi_2 \equiv (\varphi_1 \to \varphi_2) \wedge (\varphi_2 \to \varphi_1) & \text{Biimplikation}
\end{array}$$

0.5 Prädikatenlogik

0.5.1 Notation (Prädikate)
In der Aussagenlogik beinhaltet $V \cup \{ \top, \bot \}$ die atomaren Ausdrücke. In der Prädikatenlogik werden aussagenlogische Variablen aus V ersetzt durch so genannte *Prädikate* in $\mathrm{Pr}(X)$, die wiederum parametrisiert sind über eine Menge X von *Termvariablen*.

Typischerweise enthält $\mathrm{Pr}(X)$ Ausdrücke der Form $r(t_1, \ldots, t_n)$ mit $n \in \mathbb{N}^+$, wobei r zu einer anzugebenden Menge S_{rel} so genannter Relationssymbole gehört[3] (siehe § 0.6) und die t_i aus Variablen in X sowie Elementen einer ebenfalls anzugebenden Menge S_{fun} so genannter Funktionssysmbole (siehe § 0.7) zusammengesetzt werden.

Wir schreiben $\vec{x}$ für $x_1, \ldots, x_n$ und $\vec{t}$ für $t_1, \ldots, t_n$.

0.5.2 Definition (Syntax der Prädikatenlogik)
Sei X eine Menge von *Termvariablen*.
Sei $\mathrm{Pr}(X)$ eine Menge von *Prädikaten* über X.
Dann ist die *Menge* $\mathbf{P}(X)$ *der prädikatenlogischen Formeln zu* X definiert als kleinste Menge mit:

- $\mathrm{Pr}(X) \cup \{ \top, \bot \} \subseteq \mathbf{P}(X)$;
- wenn $\varphi \in \mathbf{P}(X)$,
  dann auch $\neg\varphi \in \mathbf{P}(X)$;
- wenn $\{ \varphi_1, \varphi_2 \} \subseteq \mathbf{P}(X)$,
  dann auch $\{ (\varphi_1 \wedge \varphi_2), (\varphi_1 \vee \varphi_2), (\varphi_1 \to \varphi_2), (\varphi_1 \leftrightarrow \varphi_2) \} \subseteq \mathbf{P}(X)$,
- wenn $\varphi \in \mathbf{P}(X)$,
  dann auch $\{ (\forall x.\varphi), (\exists x.\varphi) \} \subseteq \mathbf{P}(X)$.

Wir verwenden $\varphi, \psi, \ldots$ als Metavariablen für logische Formeln.

0.5.3 Notation (Quantoren und Variablen)
In den Formeln $(\forall x.\varphi)$ und $(\exists x.\varphi)$ „binden“ die Quantoren jeweils die Variable x in der Formel φ. Um Missverständnisse durch verschachtelte Mehrfachbindung zu vermeiden, benutzen wir immer paarweise verschiedene Variablen in den vorkommenden Quantoren.

Quantoren werden häufig in eingeschränkter Form verwendet:

$$\begin{array}{lcl}
(\forall x{\in}A \,.\, \varphi) & \triangleq & (\forall x \,.\, x{\in}A \to \varphi) \\
(\exists x{\in}A \,.\, \varphi) & \triangleq & (\exists x \,.\, x{\in}A \wedge \varphi)
\end{array}$$

Im Fach *Logik* gilt dabei, dass die Menge A nicht leer sein darf.

[3] Üblicherweise ist das Gleichheitssymbol $=$ in S_{rel} enthalten.

Oft werden Formeln φ zusammen mit den in ihnen frei (d.h. nicht durch Quantoren gebunden) vorkommenden Variablen $\vec{x}$ notiert als $\varphi(\vec{x})$. Das erleichtert die Notation zur Instanziierung der Variablen bzw. zum Einsetzen von konkreten Termen in entsprechender Form als $\varphi(\vec{t})$.

0.5.4 Notation (Quantoren und Klammerung)
Für Quantoren gelten folgende Regeln des *Operatorenvorrangs*:

- $\forall$ und $\exists$ binden gleich stark;
- $\forall$ und $\exists$ binden schwächer als alle anderen Konnektive.

0.5.5 Notation (Eindeutigkeit)
Die Aussage „es gibt *genau* ein x für das $\varphi(x)$ gilt" wird abgekürzt durch:

$$\exists! x\,.\,\varphi(x) \quad \triangleq \quad \Big(\exists z\,.(\varphi(z))\Big) \wedge \Big(\forall x, y\,.\,\varphi(x) \wedge \varphi(y) \to x = y\Big)$$

0.5.6 Proposition (Logische Äquivalenz (II))

$$\begin{array}{rcl} \neg(\forall x\,.\,\varphi) & \equiv & \exists x\,.\neg\varphi \\ \neg(\exists x\,.\,\varphi) & \equiv & \forall x\,.\neg\varphi \end{array}$$

0.5.7 Proposition (Mathematische Induktion)
Sei $P(n)$ ein Prädikat über den natürlichen Zahlen $\mathbb{N}$. Falls

1. $P(0)$
2. $P(n) \to P(n{+}1)$ für alle $n \in \mathbb{N}$

beide gelten (bzw. bewiesen werden können), dann gilt auch:

$$\forall n \in \mathbb{N}\,.\,P(n)$$

0.5.8 Notation (Induktionsschemata)
Sei $P(n)$ ein Prädikat über den natürlichen Zahlen $\mathbb{N}$.
Das Prinzip aus Proposition 0.5.7 kann knapp in einer einheitlichen Formel, genannt *Induktionsschema*, zusammengefasst werden.

$$\Big(\, P(0) \wedge \Big(\, \forall n \in \mathbb{N}.\big(P(n) \to P(n{+}1)\big)\,\Big)\,\Big) \to (\,\forall x \in \mathbb{N}.P(x)\,)$$

0.5.9 Proposition (Werteverlaufsinduktion)
Sei $P(n)$ ein Prädikat über den natürlichen Zahlen $\mathbb{N}$. Dann gilt:

$$\Big(\, \forall n \in \mathbb{N}.\Big(\, \big(\forall m \in [0,\, n{-}1].P(m)\big) \to P(n)\,\Big)\,\Big) \to (\,\forall x \in \mathbb{N}.P(x)\,)$$

0.5.10 Bemerkung Abgesehen von den oben definierten Induktionsprinzipien und -schemata im Fall der natürlichen Zahlen kann man Induktionsprinzipien und -schemata für jede per struktureller Induktion definierte Menge (siehe Notation 0.3.13) begründen.

0.6 Relationen

0.6.1 Definition (Kartesisches Produkt)
Seien $A_1, \ldots, A_n$ beliebige Mengen für $n \in \mathbb{N}$. Dann heißt

$$\prod_{i=1}^{n} A_i \quad \triangleq \quad \{\, (a_1,\ \ldots,\ a_n) \mid \forall i \in [1,\, n] \,.\, a_i \in A_i \,\}$$

[kartesisches] Produkt der A_i.
Es wird auch notiert als $A_1 \times \cdots \times A_n$ (siehe Def. 0.3.9 für $n = 2$).

- Die Elemente von $\prod_{i=1}^{n} A_i$ heißen n-*Tupel*.
- 2-Tupel heißen *Paare*.
- $()$ heißt *leeres Tupel*; es wird häufig auch mit λ bezeichnet.

Für eine beliebige Menge A und $n \in \mathbb{N}$ definieren wir $A^n \triangleq \prod_{i=1}^{n} A$.

0.6.2 Bemerkung (Spezialfälle)
Sei A eine beliebige Menge.

- $A^0 = \{\, () \,\}$
- $\prod_{i=1}^{n} A_i = \emptyset$, falls $A_k = \emptyset$ für mindestens ein $k \in [1, n]$.

0.6.3 Definition (Relation)
Seien $A_1, \ldots, A_n$ beliebige Mengen.
Sei $R \subseteq \prod_{i=1}^{n} A_i$. Dann heißt

$$\underbrace{R}_{\text{Graph}} : \underbrace{(A_1, \ldots, A_n)}_{\text{Typ}}$$

n-*stellige Relation*. Der Spezialfall der 2-stelligen Relation heißt auch *binäre Relation*, alternativ mit Infixschreibweise $a\ R\ b$ für $(a, b) \in R$.

Zwei Relationen $R_A : (A_1, \ldots, A_n)$ und $R_B : (B_1, \ldots, B_m)$ sind *gleich*, falls

1. $n = m$,
2. $A_i = B_i$ für alle $1 \leqslant i \leqslant n$, und
3. $R_A = R_B$.

0.6.4 Definition (Spezialfälle)
Seien A und B beliebige Mengen.

- $\emptyset_{A,B} : (A, B) \quad$ mit $\quad \emptyset_{A,B} \triangleq \emptyset$
  bezeichnet die *leere Relation* (mit Typ (A, B)).
- $\nabla_{A,B} : (A, B) \quad$ mit $\quad \nabla_{A,B} \triangleq A \times B$
  bezeichnet die *universelle Relation* bzw. *Allrelation* (mit Typ (A, B)).

Wenn der Typ implizit klar ist, kann der Index weggelassen werden.

0.6.5 Notation

- In [EMC+01] wird eine Relation *Rel* anstelle von $\mathrm{Graph}(\mathit{Rel}) : \mathrm{Typ}(\mathit{Rel})$ als Paar $\langle \mathrm{Typ}(\mathit{Rel}), \mathrm{Graph}(\mathit{Rel}) \rangle$ notiert (mit anderer Typnotation).
- Ist der Typ einer Relation implizit klar, wird oft nur R angegeben. In diesen Fällen wird auch der bloße Graph als Relation bezeichnet.

0.6.6 Definition (Totalität und Eindeutigkeit)
Sei $R : (A, B)$ eine binäre Relation. R heißt

1. *linkstotal*, falls:
$$\forall a{\in}A \,.\, \exists b{\in}B \,.\, aRb$$
2. *rechtstotal*, falls:
$$\forall b{\in}B \,.\, \exists a{\in}A \,.\, aRb$$
3. *linkseindeutig*, falls:
$$\forall a_1, a_2{\in}A \,.\, \forall b{\in}B \,.\, \big(\, a_1Rb \wedge a_2Rb \to a_1{=}a_2 \,\big)$$
4. *rechtseindeutig*, falls:
$$\forall a{\in}A \,.\, \forall b_1, b_2{\in}B \,.\, \big(\, aRb_1 \wedge aRb_2 \to b_1{=}b_2 \,\big)$$

0.6.7 Definition (Komposition) Seien A, B, C drei beliebige Mengen.
Seien $P : (A, B)$ und $Q : (B, C)$ zwei beliebige Relationen.
Dann ist die *Komposition* $P;Q : (A, C)$ definiert wie folgt:
$$P;Q \;\triangleq\; \{\, (a,\ c) \in A \times C \mid \exists b{\in}B \,.\, aPb \wedge bQc \,\}$$

0.6.8 Notation Aus Bequemlichkeit schreiben wir oft PQ anstelle von $P;Q$. Aus Gründen der Konsistenz mit dem Kompositionsbegriff für Funktionen (siehe §0.7) verwenden wir auch die Notation: $Q \circ P \triangleq P;Q$

0.6.9 Proposition (Assoziativität der Komposition)
Seien A, B, C, D beliebige Mengen.
Seien $P : (A, B)$, $Q : (B, C)$ und $R : (C, D)$ drei beliebige Relationen.
Dann gilt:
$$P;(Q;R) = (P;Q);R$$

0.6.10 Definition (Umkehrrelation)
Seien A und B zwei beliebige Mengen. Sei $R : (A, B)$.
Die *Umkehrrelation* von R, geschrieben $R^{-1} : (B, A)$, ist definiert durch:
$$R^{-1} \;\triangleq\; \{\, (b,\ a) \mid (a,\ b) \in R \,\}$$

0.6.11 Proposition (Eigenschaften der Umkehrrelation)
Seien A, B, C drei beliebige Mengen. Seien $R : (A, B)$ und $Q : (B, C)$ zwei beliebige Relationen. Dann gilt:

1. $(R^{-1})^{-1} = R$
2. $(RQ)^{-1} = Q^{-1}R^{-1}$
3. $(Q \circ R)^{-1} = R^{-1} \circ Q^{-1}$

0.7 Abbildungen

0.7.1 Definition (Partielle Abbildung)
Sei $f : (A, B)$ eine rechtseindeutige Relation. Dann heißt f *partielle Abbildung* f *vom Typ* $A \rightharpoonup B$, geschrieben $f : A \rightharpoonup B$.

1. Wir schreiben $f(a) = b$, falls $(a, b) \in f$.
2. $f(a)$ ist *Funktionswert von* f *an der Stelle* a.
3. A heißt *Argumentbereich* bzw. *Domain* von f, geschrieben $\mathrm{dom}(f)$.
   B heißt *Zielbereich* bzw. *Codomain* von f, geschrieben $\mathrm{cod}(f)$.
4. Seien $A_0 \subseteq A$ und $B_0 \subseteq B$. Dann heißen die Mengen
$$\begin{aligned} f(A_0) &\triangleq \{\, b \in B \mid \exists a \in A_0 \,.\, f(a) = b \,\} \\ f^{-1}(B_0) &\triangleq \{\, a \in A \mid \exists b \in B_0 \,.\, f(a) = b \,\} \end{aligned}$$
   Bild von A_0 *bzgl.* f, sowie *Urbild von* B_0 *bzgl.* f.
5. Die Mengen
$$\begin{aligned} \mathrm{Bild}(f) &\triangleq f(A) \\ \mathrm{Def}(f) &\triangleq f^{-1}(B) \end{aligned}$$
   heißen *Bildbereich* $\mathrm{Bild}(f) \subseteq B$ und *Definitionsbereich* $\mathrm{Def}(f) \subseteq A$.
6. Darüberhinaus heißt f *[totale] Abbildung*, geschrieben $f : A \to B$, falls f linkstotal ist.

Wir verwenden die Begriffe *Abbildung* und *Funktion* synonym.

0.7.2 Notation Anstelle von Paaren (a, b) verwendet man im Kontext von Abbildungen häufig auch die Schreibweise $a \mapsto b$. Partielle Abbildungen können dann auch durch folgende Schreibweise definiert werden:

$$f : A \rightharpoonup B; a \mapsto \mathrm{Ber}_f(a) \qquad \text{bzw.} \qquad f : A \rightharpoonup B; \begin{cases} a \mapsto \mathrm{Ber}_1(a) \,;\, P_1(a) \\ a \mapsto \mathrm{Ber}_2(a) \,;\, P_2(a) \\ \dots \end{cases}$$

wobei die Prädikate $P_1, P_2, \dots$ eine Fallunterscheidung definieren und die $\mathrm{Ber}_f, \mathrm{Ber}_1, \mathrm{Ber}_2, \dots$ jeweils „Berechnungsvorschriften“ repräsentieren.

0.7.3 Proposition (Gleichheit partieller Abbildungen)
Zwei partielle Abbildungen $f_1 : A_1 \rightharpoonup B_1$ und $f_2 : A_2 \rightharpoonup B_2$ sind *gleich*, falls $A_1 = A_2$, $B_1 = B_2$, $\mathrm{Def}(f_1) = \mathrm{Def}(f_2)$ und $\forall x \in \mathrm{Def}(f_1) \,.\, f_1(x) = f_2(x)$.

0.7.4 Theorem (Komposition partieller Abbildungen)
Seien $f : A \rightharpoonup B$ und $g : B \rightharpoonup C$ partielle Abbildungen. Dann ist deren Komposition $(g \circ f) : A \rightharpoonup C$ wieder eine partielle Abbildung.
Außerdem gilt: $(g \circ f)(x) = g(f(x))$ für alle $x \in \mathrm{Def}(g \circ f)$.

0.7.5 Definition (Eigenschaften partieller Abbildungen)

Begriff	LT	LE	RE	RT
partielle Abbildung			•	
Abbildung	•		•	
injektive partielle Abbildung		•	•	
surjektive partielle Abbildung			•	•
bijektive partielle Abbildung		•	•	•
Bijektion	•	•	•	•

0.7.6 Proposition (Umkehrabbildung I)
Sei $f : A \rightharpoonup B$ eine partielle Abbildung. Ist f injektiv, dann ist auch die Umkehrrelation $f^{-1} : (B, A)$ eine injektive partielle Abbildung.
Außerdem gilt: $(f^{-1})^{-1} = f$.

0.7.7 Definition (Isomorphie)
Seien A und B zwei Mengen. Wenn eine Bijektion $f : A \to B$ existiert, dann heißen A und B *isomorph*, geschrieben $A \cong B$.

0.7.8 Proposition (Umkehrabbildung II)
Eine Relation $f : (A, B)$ ist genau dann eine Bijektion,
wenn es eine Relation $g : (B, A)$ gibt mit: $f \circ g = \Delta_B$ und $g \circ f = \Delta_A$.
Außerdem gilt: $g = f^{-1}$.

0.7.9 Theorem (Kürzbarkeit)
Seien A, B, C, D Mengen.
Seien $f : A \to B$, $g_1, g_2 : B \to C$, und $h : C \to D$ Abbildungen.

1. Falls f surjektiv, dann gilt: $(\ g_1 \circ f = g_2 \circ f\ ) \ \to\ g_1 = g_2$
2. Falls h injektiv, dann gilt: $(\ h \circ g_1 = h \circ g_2\ ) \ \to\ g_1 = g_2$

0.7.10 Bemerkung Zu jeder binären Relation $R : (A, B)$ gibt es zwei Abbildungen f_R und g_R mit dem „gleichen Informationsgehalt“, definiert durch:

$$\begin{array}{rccl} f_R : & A & \to & 2^B \\ & a & \mapsto & \{\, b \in B \mid a\ R\ b \,\} \end{array} \qquad \begin{array}{rccl} g_R : & B & \to & 2^A \\ & b & \mapsto & \{\, a \in A \mid a\ R\ b \,\} \end{array}$$

0.8 Mengen++

0.8.1 Definition (Charakteristische Funktion)
Seien T, A zwei Mengen mit $T \subseteq A$.
Dann heißt die Abbildung $\chi_T : A \to \{0,\ 1\}$, definiert durch

$$\chi_T(a) \triangleq \begin{cases} 1 & \text{falls } a \in T \\ 0 & \text{falls } a \notin T \end{cases}$$

die *charakteristische Funktion von* T *bzgl.* A.

0.8.2 Definition (Multimengen) Sei A eine Menge. Dann bezeichnet die Abbildung $M_T : A \to \mathbb{N}$ eine eindeutig gegebene *Multimenge* T. Für jedes $a \in A$ heißt $M_T(a)$ die *Häufigkeit* bzw. *Multiplizität* von a in T.
Wir definieren die *Größe* von T als die Summe der Häufigkeiten aller Elemente von A in T:

$$\#(T) \triangleq \sum_{a \in A} M_T(a)$$

0.8.3 Definition (Mengenfamilie)
Sei M eine Menge von Mengen.
Sei I eine sogenannte *Indexmenge*.
Eine surjektive Abbildung $A : I \to M$ heißt auch *Mengenfamilie* $(A_i)_{i \in I}$.

0.8.4 Bemerkung Mengenfamilien verwendet man oft, wenn man M implizit lassen möchte.

0.9 Kardinalität

0.9.1 Definition (Größe von Mengen (II))
Eine Menge A heißt *endlich*, falls es eine Bijektion vom Typ $A \to [1, n]$ für $n \in \mathbb{N}$ gibt; dann bezeichnet $\#(A) \triangleq n$ die entsprechende Anzahl der Elemente in A. Falls ein solches n nicht existiert, heißt A *unendlich* und wir notieren $\#(A) \triangleq \infty$.

Für $n \in \mathbb{N}$ definieren wir $\infty + n = \infty$ und $\infty - n = \infty$.
Außerdem gelte: $\forall n \in \mathbb{N} \,.\, n < \infty$.

0.9.2 Proposition
Seien A und B endliche Mengen. Dann gilt:

$$\#(A \cup B) = \#(A) + \#(B) - \#(A \cap B)$$

0.9.3 Definition (Kardinalität; Vergleich von Mengengrößen)
Seien A und B zwei Mengen.

1. A und B haben die gleiche Kardinalität,
   geschrieben $\mathrm{card}(A) = \mathrm{card}(B)$,
   falls es eine *Bijektion* vom Typ $A \to B$ gibt.
   A und B heißen dann auch *äquipotent*.

2. A hat höchstens die Kardinalität von B,
   geschrieben $\mathrm{card}(A) \leqslant \mathrm{card}(B)$,
   falls es eine *injektive* Abbildung vom Typ $A \to B$ gibt.

3. A hat eine echt kleinere Kardinalität als B,
   geschrieben $\mathrm{card}(A) < \mathrm{card}(B)$,
   falls $\mathrm{card}(A) \leqslant \mathrm{card}(B)$ und $\mathrm{card}(A) \neq \mathrm{card}(B)$.

0.9.4 Notation In der Literatur werden häufig sowohl die Anzahl $\#(A)$ als auch die Kardinalität $\mathrm{card}(A)$ einer Menge A mit $|A|$ bezeichnet. Das führt gelegentlich zu Mehrdeutigkeiten bei unendlichen Mengen.

0.9.5 Definition (Abzählbarkeit) Sei A eine Menge.

1. A heißt *abzählbar*, falls A endlich oder äquipotent zu $\mathbb{N}$ ist.
2. A heißt *abzählbar unendlich*, falls A äquipotent zu $\mathbb{N}$ ist.
3. A heißt *überabzählbar*, falls $\mathrm{card}(\mathbb{N}) < \mathrm{card}(A)$.

0.9.6 Proposition (Größe von Zahlenmengen)

1. $\#(\mathbb{N}) = \#(\mathbb{Z}) = \#(\mathbb{Q}) = \#(\mathbb{R}) = \infty$
2. $\mathrm{card}(\mathbb{N}) = \mathrm{card}(\mathbb{Z}) = \mathrm{card}(\mathbb{Q})$
3. $\mathrm{card}(\mathbb{Q}) < \mathrm{card}(\mathbb{R})$

0.9.7 Theorem (Cantor)
Sei A eine Menge. Dann gilt:

$$\mathrm{card}(A) < \mathrm{card}(\mathcal{P}(A))$$

0.10 Homogene Relationen

0.10.1 Definition Sei $R : (A_1, \ldots, A_n)$ eine Relation.
R heißt *homogen*, falls $A_i = A_j$ für alle $1 \leqslant i, j \leqslant n$.
Die binäre Relation $\mathrm{Id}_A : (A, A)$ definiert durch $\mathrm{Id}_A \triangleq \{(a,\ a) \mid a{\in}A\}$ heißt *Identität auf* A. Sie wird auch *Diagonalrelation* Δ_A genannt.

0.10.2 Definition (Eigenschaften homogener Relationen)
Sei $R : (A, A)$ eine binäre homogene Relation. Dann heißt R:

reflexiv	falls	$\forall a{\in}A \,.\, aRa$	$\Delta_A \subseteq R$
irreflexiv	falls	$\forall a{\in}A \,.\, \neg(aRa)$	$\Delta_A \cap R = \emptyset$
symmetrisch	falls	$\forall a, b{\in}A \,.\, aRb \to bRa$	$R^{-1} \subseteq R$
antisymmetrisch	falls	$\forall a, b{\in}A \,.\, aRb \wedge bRa \to a{=}b$	$R^{-1} \cap R \subseteq \Delta_A$
transitiv	falls	$\forall a, b, c{\in}A \,.\, aRb \wedge bRc \to aRc$	$R \circ R \subseteq R$
linear	falls	$\forall a, b{\in}A \,.\, a{\neq}b \to (aRb \vee bRa)$	$\nabla_{A,A} \setminus \Delta_A \subseteq R^{-1} \cup R$

0.10.3 Definition Sei $R : (A, A)$ eine binäre homogene Relation. Dann heißt R^n für $n \in \mathbb{N}$ die n-fache Komposition von R, definiert durch:

$$\begin{aligned} R^0 &\triangleq \Delta_A \\ R^{n+1} &\triangleq RR^n \end{aligned}$$

0.10.4 Bemerkung Die n-fache Komposition einer Relation kann gleichwertig links- bzw rechts-induktiv definiert werden:

$$R^{n+1} \triangleq RR^n \qquad \text{oder auch} \qquad R^{n+1} \triangleq R^n R$$

([Sch03] komponiert von links, [EMC+01] komponiert von rechts.)
Je nach Kontext erlauben wir uns beide Varianten.

0.10.5 Definition (Abschluss)
Sei $R : (A, A)$ eine binäre homogene Relation.

1. Der *reflexive Abschluss* von R ist definiert durch:
$$r(R) \triangleq R \cup \Delta_A$$
2. Der *symmetrische Abschluss* von R ist definiert durch:
$$s(R) \triangleq R \cup R^{-1}$$
3. Der *transitive Abschluss* von R ist definiert durch:
$$t(R) \triangleq \bigcup_{n \in \mathbb{N}^+} R^n$$

Der Typ der Abschlussrelationen entspricht
in allen drei Fällen jeweils dem Typ der Basisrelation:
$\mathrm{Typ}(r(R)) = \mathrm{Typ}(s(R)) = \mathrm{Typ}(t(R)) = \mathrm{Typ}(R)$.

0.11 Ordnungen

0.11.1 Definition (Ordnungen)
Sei $R : (A, A)$ eine binäre homogene Relation. Die Tabelle benennt die Mindestkriterien, so dass für R der jeweilige Ordnungsbegriff gilt.

Ordnungsbegriff	reflexiv	transitiv	antisymmetrisch	linear
Quasiordnung	•	•		
partielle Ordnung	•	•	•	
totale Ordnung	•	•	•	•

0.11.2 Notation (Ordnungsbegriffe)

- Quasiordnungen heißen auch *Präordnungen*.
- Partielle Ordnungen heißen auch *Halbordnungen*.
- Totale Ordnungen heißen auch *lineare Ordnungen*.
- Irreflexive Ordnungen heißen auch *streng* oder *strikt*.

0.12 Äquivalenzen

0.12.1 Definition (Äquivalenzrelation)
Sei $R : (A, A)$ eine binäre homogene Relation.
R heißt *Äquivalenzrelation* bzw. *Äquivalenz [auf A]*,
wenn R sowohl (1) reflexiv, (2) symmetrisch, als auch (3) transitiv ist.

0.12.2 Notation
Äquivalenzen sind binäre Relationen, also Mengen (von Paaren).
Wir können daher Äquivalenzrelationen vergleichen, zum Beispiel bzgl. Mengeninklusion („$\subseteq$“) und Mengengrößen ($\#(\cdot)$).

0.12.3 Lemma
Sei A eine beliebige Menge. Dann sind eindeutig bestimmt:

- $\nabla_{A,A}$ als $\subseteq$-größte Äquivalenz auf A;
- Δ_A als $\subseteq$-kleinste Äquivalenz auf A.

0.12.4 Proposition (Erzeugte Äquivalenz)
Sei $R : (A, A)$ eine beliebige homogene Relation.
Unter allen Relationen, die Obermengen von R sind, ist:

1. $r(R)$ die $\subseteq$-kleinste reflexive Relation,
2. $s(R)$ die $\subseteq$-kleinste symmetrische Relation,
3. $t(R)$ die $\subseteq$-kleinste transitive Relation, und
4. $t(s(r(R)))$ die $\subseteq$-kleinste Äquivalenz.

0.12.5 Definition (Äquivalenzklassen)
Sei $R : (A, A)$ eine Äquivalenz. Sei $a \in A$. Dann heißt:

$$[a]_R \triangleq \{\, x \in A \mid (a, x) \in R \,\}$$

Äquivalenzklasse von a bzgl. R.

0.12.6 Notation Wenn die intendierte Äquivalenz R aus dem Kontext eindeutig hervorgeht, schreiben wir auch $[a]$ anstelle von $[a]_R$.

0.12.7 Proposition
Sei $R : (A, A)$ eine Äquivalenz. Sei $a \in A$. Dann gilt:

1. $a \in [a]$
2. $(a_1, a_2) \in R \quad$ g.d.w. $\quad [a_1] = [a_2]$
3. $(a_1, a_2) \notin R \quad$ g.d.w. $\quad [a_1] \cap [a_2] = \emptyset$

0.12.8 Definition (Divison mit Rest)
Sei $n \in \mathbb{N}$. Sei $k \in \mathbb{N}^+$. Dann gilt folgende Zerlegung:

$$\exists! m \in \mathbb{N} \ . \ \exists! r \in [0, k-1] \ . \ n = \underbrace{m}_{n \operatorname{div} k} \cdot k \quad + \underbrace{r}_{n \bmod k}$$

Wegen der Eindeutigkeit der angegebenen Zerlegung mittels m und r bezeichnen div und mod wohldefinierte Abbildungen.
Die Abbildung $n \bmod k$ wird oft auch mit $n \,\%\, k$ bezeichnet.
Wir definieren $\equiv_k \triangleq \{(n_1,\ n_2) \mid n_1 \bmod k = n_2 \bmod k\}$.

0.12.9 Definition (Quotient)
Sei $R : (A, A)$ eine Äquivalenz.
Dann heißt die Menge aller Äquivalenzklassen, gegeben durch

$$A/R \triangleq \{\ [a]_R \mid a \in A\ \}$$

der *Quotient von* A *bzgl.* R.
Die Anzahl $\#(A/R)$ der Äquivalenzklassen bzgl. R heißt *Index von* R.

0.12.10 Definition (Repräsentantensystem)
Sei $R : (A, A)$ eine Äquivalenz. Sei $S \subseteq A$.
S heißt *Repräsentantensystem von* A/R, falls gilt:

$$\forall a \in A \ . \ \exists! s \in S \ . \ s \in [a]$$

0.12.11 Proposition
Sei S *Repräsentantensystem von* A/R. Dann gilt:

1. $S \cong A/R$
2. $A = \bigcup_{x \in S} [x]_R$

0.12.12 Definition (Kern einer Abbildung)
Sei $f : A \to B$ eine Abbildung. Dann heißt die Relation

$$\operatorname{Ker}(f) \triangleq \{\ (a_1,\ a_2) \in A \times A \mid f(a_1) = f(a_2)\ \}$$

der *Kern von* f.

0.12.13 Proposition (Kern einer Abbildung)
Sei $f : A \to B$ eine Abbildung.
Dann ist $\operatorname{Ker}(f)$ eine Äquivalenz [auf A].

0.13 Faktorisierungssätze

0.13.1 Theorem (Faktorisierung I)
Sei $f : A \to B$ eine Abbildung. Dann existieren

- eine Menge C (eindeutig bestimmt „bis auf Isomorphie“),
- eine surjektive Abbildung $g : A \to C$,
- eine injektive Abbildung $h : C \to B$,

so dass $f = h \circ g$.

0.13.2 Proposition
Sei $R : (A, A)$ eine beliebige Äquivalenz. Sei

$$\begin{array}{rcccc} \mathrm{nat}_R & : & A & \to & A/R \\ & & a & \mapsto & [a]_R \end{array}$$

die so genannte *natürliche Abbildung zu* R. Dann gilt $\mathrm{Ker}(\mathrm{nat}_R) = R$.

0.13.3 Theorem (Faktorisierung II)
Sei $f : A \to B$ eine Abbildung.
Seien $\mathrm{nat}_f \triangleq \mathrm{nat}_{\mathrm{Ker}(f)}$, d.h.

$$\begin{array}{rcccc} \mathrm{nat}_f & : & A & \to & A/\mathrm{Ker}(f) \\ & & a & \mapsto & [a]_{\mathrm{Ker}(f)} \end{array}$$

die *natürliche Abbildung von* f, und

$$\begin{array}{rcccc} \mathrm{val}_f & : & A/\mathrm{Ker}(f) & \to & B \\ & & [a]_{\mathrm{Ker}(f)} & \mapsto & f(a) \end{array}$$

Es gilt:

1. nat_f ist surjektiv, val_f ist injektiv.
2. $f = \mathrm{val}_f \circ \mathrm{nat}_f$
3. $\mathrm{Bild}(f) \cong A/\mathrm{Ker}(f)$

0.13.4 Bemerkung Im Faktorisierungssatz ist
$\mathrm{Bild}(f)$ isomorph zu einem Repräsentantensystem für $A/\mathrm{Ker}(f)$.

1 Formale Sprachen

1.1 Wörter

1.1.1 Definition (Wörter)
Sei Σ eine endliche Menge ($\neq \emptyset$), genannt *Alphabet*.
Die Elemente $a_i \in \Sigma$ werden auch *Symbol*, *Zeichen*, *Buchstabe* genannt.

1. Ein *Wort* w über Σ ist eine endliche Sequenz $(a_1, \ldots, a_n)$ mit $n \in \mathbb{N}$ und $a_i \in \Sigma$ für alle $1 \leqslant i \leqslant n$.
2. Wir bezeichnen mit $|w| = n$ die Länge des Wortes $w = (a_1, \ldots, a_n)$.
3. Wir bezeichnen das *leere Wort*, d.h. $n = 0$, mit ε.
4. Wir bezeichnen mit $(w)_i$ die *Projektion* des Wortes w auf das i-te Symbol, definiert lediglich für $1 \leqslant i \leqslant |w|$.
5. Wir bezeichnen mit $|w|_a$ die Anzahl der Vorkommen von $a \in \Sigma$ in w.
6. Σ^* bezeichnet die Menge aller [endlichen] Wörter über Σ.

Häufig vereinfachen wir $(a_1, \ldots, a_n)$ zu $a_1 \cdots a_n$.

1.1.2 Definition (Operationen auf Wörtern)
Seien $v = (a_1, \ldots, a_m)$ und $w = (b_1, \ldots, b_n)$ zwei Wörter aus Σ^*.

1. Die *Konkatenation* von v und w ist definiert durch:
$$v \cdot w \triangleq (a_1, \ldots, a_m, b_1, \ldots, b_n)$$
2. v heißt *Präfix* von w, falls es $u \in \Sigma^*$ gibt, so dass $v \cdot u = w$.
3. v heißt *Suffix* von w, falls es $u \in \Sigma^*$ gibt, so dass $u \cdot v = w$.
4. v heißt *Teilwort* von w, falls es $u_1, u_2 \in \Sigma^*$ gibt, so dass $(u_1 \cdot v) \cdot u_2 = w$.

1.1.3 Notation Seien $u, v, w \in \Sigma^*$.
Anstelle von $v \cdot w$ schreiben wir oft vw.
Aufgrund der Assoziativität der Konkatenation,
also wegen $(u \cdot v) \cdot w = u \cdot (v \cdot w)$,
erlauben wir auch die klammerlose Notation $u \cdot v \cdot w$ bzw. uvw.

1.1.4 Lemma (Dekomposition) Sei w ein Wort über dem Alphabet Σ.
Dann gilt genau einer der beiden folgenden Fälle:

1. $w = \varepsilon$ (also $|w| = 0$), oder
2. es gibt $a \in \Sigma$ und $v \in \Sigma^*$ mit $|v| = |w| - 1$, so dass $w = av$.

Die obige Dekomposition „von links“ entspricht den in der Informatik üblichen Listenstrukturen. Alternativ – und völlig gleichwertig – kann man die Dekomposition auch „von rechts“ ausführen. In jedem Fall läßt sich damit ein Induktionsprinzip herleiten.

$$\left(P(\varepsilon) \wedge \left(\forall v \in \Sigma^*. \big(P(v) \to \forall a \in \Sigma \;.\; P(av) \big) \right) \right) \to \big(\forall w \in \Sigma^*. P(w) \big)$$

$$\left(P(\varepsilon) \wedge \left(\forall v \in \Sigma^*. \big(P(v) \to \forall a \in \Sigma \;.\; P(va) \big) \right) \right) \to \big(\forall w \in \Sigma^*. P(w) \big)$$

Ebenso wird häufig das Prinzip der Induktion über die Wortlänge verwendet; es unterscheidet sich dann nicht von der natürlichen Induktion.

1.1.5 Definition (Ordnungen auf Wörtern)
Sei Σ ein Alphabet.
Sei $R : (\Sigma, \Sigma)$ eine Halbordnung.

1. Die *lexikographische Ordnung* $\ll_R : (\Sigma^*, \Sigma^*)$
   ist definiert als die kleinste Relation diesen Typs mit:
   (a) $\varepsilon \ll_R w$ für beliebiges $w \in \Sigma^*$;
   (b) wenn $(a, b) \in R$ und $a \neq b$,
   dann gilt $av \ll_R bw$ für alle $v, w \in \Sigma^*$;
   (c) wenn $v \ll_R w$ für $v, w \in \Sigma^*$,
   dann gilt $av \ll_R aw$ für alle $a \in \Sigma$.
2. Die *Standardordnung* $\ll_R^S : (\Sigma^*, \Sigma^*)$ ist definiert durch:

$$\ll_R^S \triangleq \{\ (v,\ w) \in \Sigma^* \times \Sigma^*\ \mid\ |v|<|w|\ \vee\ (\ |v|=|w| \wedge v \ll_R w\ )\ \}$$

1.1.6 Proposition
Sei Σ ein Alphabet.
Sei $R : (\Sigma, \Sigma)$ eine totale Ordnung.
Dann sind auch $\ll_R$ und $\ll_R^S$ totale Ordnungen.

1.1.7 Proposition
Sei Σ ein Alphabet.
Dann ist Σ^* abzählbar unendlich.

1.2 Sprachen

1.2.1 Definition (Sprachen)
Sei Σ ein Alphabet.
Dann heißt jede Teilmenge $A \subseteq \Sigma^*$ *Sprache über* Σ.

1.2.2 Proposition
Sei Σ ein Alphabet.
Die Menge $\mathcal{P}(\Sigma^*)$ der Sprachen über Σ ist überabzählbar.

1.2.3 Definition (Konkatenation von Sprachen)
Seien A und B Sprachen über dem Alphabet Σ.
Die Konkatenation $A \cdot B$ (oder kurz: AB) ist definiert durch:

$$A \cdot B \triangleq \{\ w \in \Sigma^* \mid \exists w_1 \in A,\ w_2 \in B\,.\, w = w_1 \cdot w_2\ \}$$

Die n*-fache Konkatenation* ist definiert durch:

$$\begin{array}{rcl@{\qquad\qquad}rcl} A^0 & \triangleq & \{\ \varepsilon\ \} & A^* & \triangleq & \bigcup_{n\geqslant 0} A^n \\ A^{n+1} & \triangleq & A \cdot A^n & A^+ & \triangleq & \bigcup_{n\geqslant 1} A^n \end{array}$$

1.2.4 Definition (Spiegelsprache)
Sei Σ ein Alphabet und $A \subseteq \Sigma^*$.
Sei $(\cdot)^R : \Sigma^* \to \Sigma^*$ definiert durch:

$$\varepsilon^R = \varepsilon \text{ und} \tag{1}$$

$$(wa)^R = a\left(w^R\right), \text{ für alle } w \in \Sigma^* \text{ und } a \in \Sigma. \tag{2}$$

Dann heißt $A^R \triangleq \{\ w^R \mid w \in A\ \}$ *Spiegelsprache von* A.

1.2.5 Lemma
Sei Σ ein Alphabet und $A \subseteq \Sigma^*$.
Dann gilt $(A^R)^R = A$.

1.2.6 Definition (Reguläre Ausdrücke)
Sei Σ ein Alphabet mit $\Sigma \cap \{\mathbf{0},\ \boldsymbol{\epsilon}\} = \emptyset$.
Die Menge $\mathcal{A}^\Sigma$ der *regulären Ausdrücke über* Σ
ist als kleinste Menge definiert,
die die folgenden $\in$-Regeln erfüllt:

1. $\mathbf{0} \in \mathcal{A}^\Sigma$, $\boldsymbol{\epsilon} \in \mathcal{A}^\Sigma$, $a \in \mathcal{A}^\Sigma$ für alle $a \in \Sigma$
2. falls $e \in \mathcal{A}^\Sigma$, dann auch $e^* \in \mathcal{A}^\Sigma$
3. falls $e_1 \in \mathcal{A}^\Sigma$ und $e_2 \in \mathcal{A}^\Sigma$, dann auch $(e_1 \cdot e_2) \in \mathcal{A}^\Sigma$
4. falls $e_1 \in \mathcal{A}^\Sigma$ und $e_2 \in \mathcal{A}^\Sigma$, dann auch $(e_1 + e_2) \in \mathcal{A}^\Sigma$

1.2.7 Notation
Der Ausdruck $e_1 \cdot e_2$ wird oft durch $e_1 e_2$ abgekürzt.
Analog zu 0.4.2 können manche Klammern weggelassen werden,
wobei gilt:

- * hat Vorrang vor $\cdot$
- $\cdot$ hat Vorrang vor $+$

1.2.8 Definition (Sprache regulärer Ausdrücke)
Sei Σ ein Alphabet mit $\Sigma \cap \{\mathbf{0},\ \boldsymbol{\epsilon}\} = \emptyset$.
Die Abbildung $L : \mathcal{A}^\Sigma \to \mathcal{P}(\Sigma^*)$ gegeben durch:

$$
\begin{array}{lcll}
\mathbf{0} & \mapsto & \emptyset & \\
\boldsymbol{\epsilon} & \mapsto & \{\varepsilon\} & \\
a & \mapsto & \{a\} & \text{für alle } a \in \Sigma \\
e^* & \mapsto & L(e)^* & \\
e_1 \cdot e_2 & \mapsto & L(e_1) \cdot L(e_2) & \\
e_1 + e_2 & \mapsto & L(e_1) \cup L(e_2) &
\end{array}
$$

definiert die zugehörige Sprache eines regulären Ausdrucks.

1.2.9 Notation
Die Schreibweisen a^n und a^* werden häufig verwechselt.
a^n steht für das einzige Wort in der Sprache $\{a\}^n$.
a^n bezeichnet somit ein einzelnes Wort.
a^* bezeichnet einen regulären Ausdruck (mit Sprache $\{\ a^n \mid n \in \mathbb{N}\ \}$).

1.3 Grammatiken

1.3.1 Definition (Grammatik)
Eine Grammatik ist ein 4-Tupel $G \triangleq (V, \Sigma, P, S)$.

- V ist ein Alphabet von *Nichtterminalsymbolen*,
  auch *Variablen* genannt.

- Σ ist ein Alphabet von *Terminalsymbolen*.
  Σ (auch Σ^*) und V müssen disjunkt sein ($V \cap \Sigma^* = \emptyset$).

- $P \subseteq \big((V \cup \Sigma)^+ \setminus \Sigma^+\big) \times (V \cup \Sigma)^*$
  ist eine endliche Menge von *Produktionen*.
- $S \in V$ ist das *Startsymbol*.

Wir benutzen $\alpha, \beta, \gamma, \eta, \pi, \ldots$ für Elemente von $(V \cup \Sigma)^*$.
Aussagen $(\pi, \pi') \in P$ schreiben wir auch als $\pi \to_G \pi'$;
wenn G im Kontext eindeutig ist, erlauben wir auch $\pi \to \pi'$.

1.3.2 Definition (Ableitung) Sei $G \triangleq (V, \Sigma, P, S)$ eine Grammatik.

1. Die Relation $\Rightarrow_G : \big((V \cup \Sigma)^*, (V \cup \Sigma)^*\big)$ ist gegeben durch:

$$\{\ (\alpha \cdot \pi \cdot \eta,\ \alpha \cdot \pi' \cdot \eta) \mid \pi \to_G \pi' \wedge \alpha,\ \eta \in (V \cup \Sigma)^*\ \} \tag{1}$$

   Sie wird als *direkte Ableitung in* G bezeichnet.

   Für $\sigma \Rightarrow_G \sigma'$ heißt σ' *in* G *direkt aus* σ *ableitbar*.

2. Die Relation $\Rightarrow_G^*$ ist der reflexiv-transitive Abschluss von $\Rightarrow_G$.
   Eine *Ableitung in* G ist eine endliche Sequenz $(\sigma_i)_{i \in [0, n]}$ mit $n \in \mathbb{N}$
   von Elementen aus $(V \cup \Sigma)^*$, so daß $\forall i \in [0, n-1] \,.\, \sigma_i \Rightarrow_G \sigma_{i+1}$.
   Wir schreiben dafür auch $\sigma_0 \Rightarrow_G \sigma_1 \Rightarrow_G \ldots \Rightarrow_G \sigma_n$.

3. Eine Ableitung $(\sigma_i)_{i \in [0, n]}$ heißt *Linksableitung*,
   wenn für jeden direkten Ableitungsschritt $\sigma_i \Rightarrow_G \sigma_{i+1}$
   mit $\sigma_i = \alpha \cdot \pi \cdot \eta$ gemäß (1) gilt, dass $\alpha \in \Sigma^*$.

4. Eine Ableitung $(\sigma_i)_{i \in [0, n]}$ heißt *Rechtsableitung*,
   wenn für jeden direkten Ableitungsschritt $\sigma_i \Rightarrow_G \sigma_{i+1}$
   mit $\sigma_i = \alpha \cdot \pi \cdot \eta$ gemäß (1) gilt, dass $\eta \in \Sigma^*$.

1.3.3 Definition (Spracherzeugung) Sei $G = (V, \Sigma, P, S)$ eine Grammatik.

1. Ein Wort $\sigma \in (V \cup \Sigma)^*$ ist *ableitbar* in G gdw $S \Rightarrow_G^* \sigma$.
   σ heißt dann auch *Satzform von* G.
2. Die von G *erzeugte Sprache* $L(G)$ ist definiert durch:

$$L(G) \triangleq \{\ w \in \Sigma^* \mid S \Rightarrow_G^* w\ \}$$

1.3.4 Definition (Äquivalenz von Grammatiken)
Seien $G_1 = (V_1, \Sigma, P_1, S_1)$ und $G_2 = (V_2, \Sigma, P_2, S_2)$
zwei Grammatiken mit demselben Terminalalphabet.
G_1 und G_2 heißen äquivalent, wenn $L(G_1) = L(G_2)$.

1.4 Chomsky-Hierarchie

1.4.1 Definition (Wortproblem)
Sei $G = (V, \Sigma, P, S)$ eine Grammatik.
Das *Wortproblem* für G besteht darin, herauszufinden
ob für ein beliebiges Wort $w \in \Sigma^*$ die Eigenschaft $w \in L(G)$ gilt.

1.4.2 Definition (Grammatik-Typen) Sei $G = (V, \Sigma, P, S)$ eine Grammatik.

- G ist *vom Typ 0*.
- G ist *vom Typ 1* bzw. *kontextsensitiv*, falls G vom Typ 0 ist und zusätzlich für jede Produktion $\pi \to_G \pi'$ gilt:

$$|\pi| \leqslant |\pi|'$$

- G ist *vom Typ 2* bzw. *kontextfrei*, falls G vom Typ 1 ist und zusätzlich für jede Produktion $\pi \to_G \pi'$ gilt:

$$\pi \in V$$

- G ist *vom Typ 3* bzw. *regulär*, falls G vom Typ 2 ist und zusätzlich für jede Produktion $\pi \to_G \pi'$ gilt:

$$\pi' \in \Sigma \cup \Sigma V$$

  In dieser Form heißt G auch *rechtslinear*.
  Ersetzt man $\Sigma \cup \Sigma V$ durch $\Sigma \cup V\Sigma$, dann heißt G *linkslinear*.

Ist $\varepsilon \in L(G)$ gewünscht, so wird bei den Typen 1, 2 und 3 zusätzlich die Produktion

$$S \to_G \varepsilon$$

erlaubt, wenn für jede Produktion $\pi \to_G \pi'$ zudem gilt:

$$\pi' \in \big(\Sigma \cup (V \backslash \{S\})\big)^*$$

1.4.3 Definition (Sprach-Typen)
Eine *Sprache* ist *vom Typ* i, falls sie von einer Grammatik des Typs i erzeugt wird. Die Menge $\mathcal{L}_i$ aller Sprachen vom Typ i ist definiert durch:

$$\mathcal{L}_i = \{\, L(G) \mid G \text{ ist vom Typ } i \,\}.$$

1.4.4 Theorem (Chomsky Hierarchie)

$$\mathcal{L}_3 \subset \mathcal{L}_2 \subset \mathcal{L}_1 \subset \mathcal{L}_0$$

1.4.5 Theorem
Sei Σ ein Alphabet und $A \subseteq \Sigma^*$.
Die folgenden Aussagen sind äquivalent.

1. Es gibt einen regulären Ausdruck $e \in \mathcal{A}^\Sigma$ mit $L(e) = A$.
2. Es gibt eine Typ-3-Grammatik G mit $L(G) = A$.

2 Reguläre Sprachen und Automaten

2.1 Deterministische endliche Automaten

2.1.1 Definition Ein *deterministischer endlicher Automat (DFA)* ist ein 5-Tupel

$$M = (Q, \Sigma, \delta, q_0, F)$$

wobei:

- Q ist eine endliche Menge von *Zuständen*,
- Σ ist eine endliche Symbolmenge, das *Eingabealphabet*,
- $\delta : (Q \times \Sigma) \to Q$ ist eine Abbildung, die *Übergangsfunktion*,
- $q_0 \in Q$ ist der *Startzustand*,
- $F \subseteq Q$ ist eine Menge von *[akzeptierenden] Endzuständen*.

2.1.2 Notation Ein Automat $(Q, \Sigma, \delta, q_0, F)$ kann auch definiert werden, indem man zu Q und Σ noch eine der folgenden Beschreibungen dazufügt:

- ein *Graph*:
  1. für jeden Zustand $q \in Q$
     zeichnen wir genau einen Knoten im Graphen (q eingekreist).
  2. Für jede Transition $\delta(q, s) = q'$
     zeichnen wir einen Pfeil von q nach q', der mit s beschriftet ist.
     Gibt es mehrere verschiedene Transitionen zwischen q und q', jeweils beschriftet mit einem der Symbole $s_1, \ldots, s_m$ ($\in \Sigma$), so benutzen wir einen einzigen Pfeil, beschriftet mit der gesamten Sequenz $s_1, \ldots, s_m$.
  3. Der Anfangszustand q_0
     wird durch einen quellenlosen Pfeil markiert, dessen Spitze auf q_0 zeigt.
  4. Jeder Endzustand $q \in F$
     wird graphisch durch einen Doppelkreis dargestellt.

- eine *Tabelle* der Form:

δ	s_1	$\cdots$	s_i	$\cdots$	s_n
q_1	$\delta(q_1, s_1)$	$\cdots$	$\delta(q_1, s_i)$	$\cdots$	$\delta(q_1, s_n)$
$\vdots$	$\vdots$		$\vdots$		$\vdots$
$F\, q_j$	$\delta(q_j, s_1)$	$\cdots$	$\delta(q_j, s_i)$	$\cdots$	$\delta(q_j, s_n)$
$\vdots$	$\vdots$		$\vdots$		$\vdots$
$SF\, q_m$	$\delta(q_m, s_1)$	$\cdots$	$\delta(q_m, s_i)$	$\cdots$	$\delta(q_m, s_n)$

  wobei S den Startzustand und F die Endzustände markiert.

2.1.3 Definition (Berechnungen via Relationen)
Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA.
Eine *Konfiguration* ist ein Paar (q, w), wobei $q \in Q$ und $w \in \Sigma^*$.
Ein *Berechnungsschritt* ist ein Element
der binären Relation $\vdash_M$ auf Konfigurationen:

$$(q, sw) \vdash_M (q', w) \quad \text{gdw} \quad \delta(q, s) = q'$$

für $q, q' \in Q, s \in \Sigma, w \in \Sigma^*$.
Eine *Berechnung* ist eine endliche Sequenz von Berechnungsschritten:

$$(q_1, w_1) \vdash_M \ldots \vdash_M (q_m, w_m) \qquad \text{bzw.} \qquad (q_1, w_1) \vdash_M^* (q_m, w_m)$$

wobei $\vdash_M^*$ den reflexiv-transitiven Abschluss von $\vdash_M$ bezeichnet.
Wir schreiben oft $\vdash$ (und $\vdash^*$) anstelle von $\vdash_M$ (und $\vdash_M^*$).
Eine Konfiguration (q, w) heißt

- *Anfangskonfiguration* zur Eingabe w, wenn $q = q_0$;
- *Folgekonfiguration* von (p, v), falls $(p, v) \vdash (q, w)$;
- *Endkonfiguration*, wenn es keine Folgekonfiguration gibt;
- *akzeptierende Konfiguration*, wenn $w = \varepsilon$ und $q \in F$.

2.1.4 Definition (Berechnungen via Abbildungen)
Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA.
Wir erweitern die Übergangsfunktion δ auf $\widehat{\delta} : Q \times \Sigma^* \to Q$ durch:

$$\begin{array}{rcl} \widehat{\delta}(q, \varepsilon) & \triangleq & q \\ \widehat{\delta}(q, ws) & \triangleq & \delta(\widehat{\delta}(q, w), s) \end{array}$$

für $q \in Q$, $w \in \Sigma^*$, $s \in \Sigma$.

2.1.5 Lemma
Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA, $q, q' \in Q$ und $w \in \Sigma^*$.
Dann gilt:

$$(q, w) \vdash^* (q', \varepsilon) \quad \text{gdw} \quad \widehat{\delta}(q, w) = q'$$

2.1.6 Definition (Akzeptierte Sprache eines DFA)
Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA.

1. M *akzeptiert* das Wort $w \in \Sigma^*$ gdw $\widehat{\delta}(q_0, w) \in F$.
   Im gegenteiligen Fall *verwirft* M das Wort w.
2. Die *von* M *akzeptierte Sprache* ist definiert durch:

$$L(M) \triangleq \{\, w \in \Sigma^* \mid \widehat{\delta}(q_0,\ w) \in F \,\}$$

2.1.7 Theorem Sei $A \subseteq \Sigma^*$ eine Sprache über Σ.
Die beiden folgenden Aussagen sind äquivalent:

1. Es existiert ein DFA $M = (Q, \Sigma, \delta, q_0, F)$ mit $L(M) = A$.
2. Es existiert eine Typ3-Grammatik $G = (V, \Sigma, P, S)$ mit $L(G) = A$.

2.2 Nichtdeterministische endliche Automaten

2.2.1 Definition (NFA)
Ein nichtdeterministischer endlicher Automat (NFA) ist ein 5-Tupel

$$M = (Q, \Sigma, \Delta, S, F)$$

wobei

- Q ist eine endliche Menge von *Zuständen*,
- Σ ist eine endliche Symbolmenge, das *Eingabealphabet*,
- $\Delta : (Q \times \Sigma) \to \mathcal{P}(Q)$ ist eine Abbildung, die *Übergangsfunktion*,
- $S \subseteq Q$ ist eine Menge von *Startzuständen*,
- $F \subseteq Q$ ist eine Menge von *[akzeptierenden] Endzuständen*.

2.2.2 Definition (Berechnungen via Relationen)
Sei $M = (Q, \Sigma, \Delta, S, F)$ ein NFA.
Konfigurationen und *Berechnungen* sind definiert wie im Fall der DFA, außer dass individuelle *Berechnungsschritte* nun definiert sind durch:

$$(q, sw) \vdash (q', w) \quad \text{gdw} \quad \Delta(q, s) \ni q'$$

für $q, q' \in Q, s \in \Sigma, w \in \Sigma^*$.

2.2.3 Definition (Berechnungen via Abbildungen)
Sei $M = (Q, \Sigma, \Delta, S, F)$ ein NFA.
Wir erweitern $\Delta : (Q \times \Sigma) \to \mathcal{P}(Q)$ auf $\widehat{\Delta} : (\mathcal{P}(Q) \times \Sigma^*) \to \mathcal{P}(Q)$ durch:

$$\widehat{\Delta}(X, \varepsilon) \triangleq X \qquad \widehat{\Delta}(X, wa) \triangleq \bigcup_{q \in \widehat{\Delta}(X,w)} \Delta(q, a)$$

Alternativ dazu verwenden wir auch:

$$\widehat{\Delta}(X, aw) \triangleq \bigcup_{q \in X} \widehat{\Delta}(\Delta(q, a), w)$$

2.2.4 Lemma
Sei $M = (Q, \Sigma, \Delta, S, F)$ ein NFA, $q, q' \in Q$ und $w \in \Sigma^*$.
Dann gilt:

$$(q, w) \vdash^* (q', \varepsilon) \quad \text{gdw} \quad \widehat{\Delta}(\{q\}, w) \ni q'$$

2.2.5 Bemerkung
Jeder DFA $(Q, \Sigma, \delta, q_0, F)$ kann via

$$\Delta(q, s) \triangleq \{\, \delta(q,\ s) \,\} \qquad \text{für alle } q \in Q, s \in \Sigma$$

auch im Format eines NFA $(Q, \Sigma, \Delta, \{q_0\}, F)$ dargestellt werden.
Beide Varianten akzeptieren die gleiche Sprache.

2.2.6 Definition (Akzeptierte Sprache eines NFA)
Sei $M = (Q, \Sigma, \Delta, S, F)$ ein NFA.

1. M *akzeptiert* das Wort $w \in \Sigma^*$
   gdw es $q_0 \in S$ und $q \in F$ gibt, so dass $(q_0, w) \vdash^* (q, \varepsilon)$.
   Im gegenteiligen Fall *verwirft* M das Wort w.
2. Die *von* M *akzeptierte Sprache* ist definiert durch:

$$L(M) \triangleq \{\, w \in \Sigma^* \mid \exists q_0 \in S \,.\, \exists q \in F \,.\, (q_0,\ w) \vdash^* (q,\ \varepsilon) \,\}$$

2.2.7 Bemerkung Die Akzeptanz des Wortes w durch Automat M kann äquivalent auch durch $\widehat{\Delta}(S, w) \cap F \neq \emptyset$ beschrieben werden.

2.2.8 Definition (Erreichbarkeit) Zu jedem endlichen Automat M enthält die Variante $\mathrm{Reach}(M)$ nur erreichbare Zustände.

Für DFA $M = (Q, \Sigma, \delta, q_0, F)$ definiere $\mathrm{Reach}(M) \triangleq (Q', \Sigma, \delta', q_0, F)$ mit Q' und $\delta' : (Q' \times \Sigma) \to Q'$ wie folgt:

$$\begin{aligned} Q' &\triangleq \{\, q \in Q \mid \exists w \in \Sigma^* \,.\, (q_0,\ w) \vdash^* (q,\ \varepsilon) \,\} \\ \delta'(q, s) &\triangleq \delta(q, s) \end{aligned}$$

Für NFA $M \triangleq (Q, \Sigma, \Delta, S, F)$ und $\mathrm{Reach}(M) \triangleq (Q', \Sigma, \Delta', S, F)$ sind Q' und $\Delta' : (Q' \times \Sigma) \to \mathcal{P}(Q')$ analog definiert.

2.2.9 Definition (Untermengen-Konstruktion)
Die Determinisierungsfunktion D zur Umwandlung eines beliebigen NFA in einen gleichwertigen DFA ist definiert wie folgt:

$$\begin{aligned} D : \quad & \mathrm{NFA} \to \mathrm{DFA} \\ & (Q, \Sigma, \Delta, S, F) \mapsto (Q_D, \Sigma, \delta, q_D, F_D) \end{aligned}$$

wobei

$$\begin{aligned} Q_D &\triangleq \mathcal{P}(Q) \\ \delta(X, a) &\triangleq \textstyle\bigcup_{q \in X} \Delta(q, a) \\ q_D &\triangleq S \\ F_D &\triangleq \{\, X \subseteq Q \mid X \cap F \neq \emptyset \,\} \end{aligned}$$

2.2.10 Lemma
Sei $M = (Q, \Sigma, \Delta, S, F)$ ein NFA. Sei $D(M) = (\ldots, \delta, \ldots)$.
Dann gilt für alle $w \in \Sigma^*, X \subseteq Q \,.\, \widehat{\delta}(X, w) = \widehat{\Delta}(X, w)$.

2.2.11 Theorem Sei M ein NFA. Dann gilt: $L(M) = L(D(M))$.

Die Untermengen-Konstruktion der Determinisierungsfunktion D generiert in der Regel einen DFA mit einer großen Zahl unerreichbarer und daher nutzloser Zustände. Daher gehen wir praktischerweise eher so vor, dass wir — schrittweise — ausgehend vom Startzustand $q_D = S$ nur die daraus erreichbaren Zustände generieren. Das Ergebnis ist $\mathrm{Reach}(D(M))$.

2.2.12 Korollar Sei $A \subseteq \Sigma^*$ eine Sprache über Σ.
Die beiden folgenden Aussagen sind äquivalent:

1. Es existiert ein NFA $M = (Q, \Sigma, \Delta, S, F)$ mit $L(M) = A$.
2. Es existiert eine Typ3-Grammatik $G = (V, \Sigma, P, S)$ mit $L(G) = A$.

2.3 Minimierung

Sei $(\phi \veebar \psi)$ genau dann, wenn $\neg(\phi \leftrightarrow \psi)$;
es bedeutet dann *entweder* ϕ *oder* ψ, *aber nicht beide*.

2.3.1 Definition (Minimalität von DFA)
Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA mit $\mathrm{Reach}(M) = M$.
M heißt *minimal*, wenn es
für alle $p, q \in Q$ mit $p \neq q$ ein $z \in \Sigma^*$ gibt, so dass gilt:

$$\widehat{\delta}(p, z) \in F \quad \veebar \quad \widehat{\delta}(q, z) \in F$$

2.3.2 Definition (Äquivalenz auf DFA-Zuständen)
Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA mit $\mathrm{Reach}(M) = M$.
Wir definieren die Relation $\equiv_M : (Q, Q)$ durch:

$$\equiv_M \quad \triangleq \quad \{\, (p,\ q) \mid \forall z \in \Sigma^* \ .\ \widehat{\delta}(p,\ z) \in F \ \Leftrightarrow\ \widehat{\delta}(q,\ z) \in F \,\}$$

2.3.3 Lemma
Die Relation $\equiv_M$ aus Definition 2.3.2 ist eine Äquivalenz.

2.3.4 Definition (Quotientenautomat)
Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA mit $\mathrm{Reach}(M) = M$.
Der Automat

$$M' = (Q', \Sigma, \delta', q_0', F')$$

definiert durch

$$\begin{aligned} Q' &\triangleq Q/\equiv_M \\ \delta'([q]_{\equiv_M}, a) &\triangleq [\delta(q, a)]_{\equiv_M} \\ q_0' &\triangleq [q_0]_{\equiv_M} \\ F' &\triangleq \{\, [q]_{\equiv_M} \mid q \in F \,\} \end{aligned}$$

2.3.5 Proposition
Der Automat M' aus Definition 2.3.4 ist minimal.
Es gilt $L(M) = L(M')$.

2.3.6 Definition (Table-Filling-Algorithmus)
Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA mit $\mathrm{Reach}(M) = M$.
Der folgende Algorithmus berechnet äquivalente Zustände von M,
die zu verschmelzen sind, um einen Minimalautomaten zu erhalten.

1. Stelle eine Tabelle aller ungeordneten Zustandspaare $\{p,\ q\}$ auf.
2. Markiere alle „Paare“ $\{p,\ q\}$ mit $p \in F$ und $q \notin F$ (oder umgekehrt).
3. Gibt es ein noch unmarkiertes „Paar“ $\{p,\ q\}$ und ein $a \in \Sigma$ derart, dass $\{\, \delta(p,\ a),\ \delta(q,\ a) \,\}$ markiert ist, so markiere auch $\{p,\ q\}$.
4. Wiederhole Schritt 3, bis sich keine neue Markierung mehr ergibt.
5. Alle jetzt noch unmarkierten „Paare“ sind verschmelzbar.

2.3.7 Lemma
Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA mit $\text{Reach}(M) = M$.
Dann gilt:

$$\equiv_M \; = \{ (q_1,\ q_2) \mid \{ q_1,\ q_2 \} \text{ ist gemäß 2.3.6(5) unmarkiert} \}$$

2.3.8 Definition (Spiegelautomat)
Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA mit $\text{Reach}(M) = M$.
Der Automat

$$M^R = (Q^R, \Sigma, \delta^R, q_0^R, F^R)$$

definiert durch:

$$\begin{aligned} Q^R &\triangleq \mathcal{P}(Q) \\ \delta^R(X, a) &\triangleq \{ q \in Q \mid \delta(q,\ a) \in X \} \\ q_0^R &\triangleq F \\ F^R &\triangleq \{ X \subseteq Q \mid q_0 \in X \} \end{aligned}$$

2.3.9 Proposition
Der Automat M aus Definition 2.3.8 ist minimal.
Es gilt $L\left(M^R\right) = (L(M))^R$.

2.3.10 Korollar
Sei M ein DFA mit $\text{Reach}(M) = M$.
Dann ist $(M^R)^R$ minimal mit $L\left((M^R)^R\right) = L(M)$.

2.3.11 Definition (Brzozowski-Suffixe)
Sei Σ ein Alphabet, $A \subseteq \Sigma^*$, und $u \in \Sigma^*$.
Dann nennen wir

$$u^{-1}A \triangleq \{ z \in \Sigma^* \mid uz \in A \}$$

die Menge der *Brzozowski-Suffixe* von u in Bezug auf A.
Wir definieren die Relation $\overset{B}{\equiv}_A : (\Sigma^*, \Sigma^*)$ durch:

$$\overset{B}{\equiv}_A \triangleq \{ (x,\ y) \mid x^{-1}A = y^{-1}A \}$$

2.3.12 Definition (Myhill-Nerode-Relation)
Sei Σ ein Alphabet und $A \subseteq \Sigma^*$.
Wir definieren die Relation $\overset{MN}{\equiv}_A : (\Sigma^*, \Sigma^*)$ durch:

$$\overset{MN}{\equiv}_A \triangleq \{ (x,\ y) \mid \forall z \in \Sigma^* \ .\ (xz \in A \leftrightarrow yz \in A) \}$$

2.3.13 Lemma (Myhill-Nerode-Relation)
Sei Σ ein Alphabet und $A \subseteq \Sigma^*$. Dann gilt:

$$\overset{B}{\equiv}_A = \overset{MN}{\equiv}_A$$

Wir verwenden $\equiv_A$ als vereinfachte Schreibweise.

2.3.14 Lemma
Die Relation $\equiv_A$ aus Lemma 2.3.13 ist eine Äquivalenz.

2.3.15 Definition (Äquivalenzklassenautomat)
Sei Σ ein Alphabet und $A \subseteq \Sigma^*$.
Sei der Index von $\equiv_A$ endlich.
Dann definieren wir den *A-Äquivalenzklassenautomaten*

$$M = (Q, \Sigma, \delta, q_0, F)$$

durch:

$$\begin{aligned} Q &\triangleq \Sigma^*/\equiv_A \\ \delta([x]_{\equiv_A}, a) &\triangleq [xa]_{\equiv_A} \\ q_0 &\triangleq [\varepsilon]_{\equiv_A} \\ F &\triangleq \{\, [x]_{\equiv_A} \mid x \in A \,\} \end{aligned}$$

2.3.16 Proposition
Der Automat M aus Definition 2.3.15 ist minimal.
Es gilt $L(M) = A$.

2.4 Eigenschaften regulärer Sprachen

2.4.1 Theorem (Myhill-Nerode)
Sei Σ ein Alphabet und $A \subseteq \Sigma^*$.
A ist genau dann regulär, wenn der Index von $\equiv_A$ endlich ist.

2.4.2 Lemma (Pumping)
Sei Σ ein Alphabet und $A \subseteq \Sigma^*$.
Sei die Formel **PUMP-REG**(A) definiert durch:

$$\textbf{PUMP-REG}(A) \triangleq \exists n \in \mathbb{N} \,.\, \forall w \in A \,.\, \Big(✪ \rightarrow \big(\exists x, y, z \in \Sigma^* \,.\, ❶ \wedge ❷ \wedge ❸ \wedge ❹ \big) \Big)$$

mit: $✪ \triangleq |w| \geqslant n$ $\quad$ $❶ \triangleq w = xyz$ $\quad$ $❷ \triangleq y \neq \varepsilon$ $\quad$ $❸ \triangleq |xy| \leqslant n$ $\quad$ $❹ \triangleq \forall k \in \mathbb{N} \,.\, xy^k z \in A$

Dann gilt:

1. Wenn A regulär ist, dann gilt **PUMP-REG**(A).
2. Wenn $\neg$ **PUMP-REG**(A) gilt, dann ist A nicht regulär.

$$\neg \textbf{PUMP-REG}(A) \equiv \forall n \in \mathbb{N} \,.\, \exists w \in A \,.\, \Big(✪ \wedge \big(\forall x, y, z \in \Sigma^* \,.\, (❶ \wedge ❷ \wedge ❸) \rightarrow \neg ❹ \big) \Big)$$
$$\neg ❹ \equiv \exists k \in \mathbb{N} \,.\, xy^k z \notin A$$

2.4.3 Lemma
Seien $M_1 = (Q_1, \Sigma, \delta_1, s_1, F_1)$ und $M_2 = (Q_2, \Sigma, \delta_2, s_2, F_2)$ DFA.
Seien:

$$\begin{aligned} Q_{1\times 2} &\triangleq Q_1 \times Q_2 \\ \delta_{1\times 2}((q_1, q_2), a) &\triangleq (\delta_1(q_1, a), \delta_2(q_2, a)) \\ s_{1\times 2} &\triangleq (s_1, s_2) \end{aligned}$$

Dann gilt :

$$\forall x \in \Sigma^* . \widehat{\delta_{1\times 2}}((p,q),x) = (\widehat{\delta_1}(p,x), \widehat{\delta_2}(q,x))$$

Der [Produkt-]Automat $M_1 \otimes M_2 \triangleq (Q_{1\times 2}, \Sigma, \delta_{1\times 2}, s_{1\times 2}, F_1 \times F_2)$
akzeptiert die Sprache $L(M_1 \otimes M_2) = L(M_1) \cap L(M_2)$.
Der [Summen-]Automat $M_1 \oplus M_2 \triangleq (Q_{1\times 2}, \Sigma, \delta_{1\times 2}, s_{1\times 2}, (F_1 \times Q_2) \cup (Q_1 \times F_2))$
akzeptiert die Sprache $L(M_1 \oplus M_2) = L(M_1) \cup L(M_2)$.

2.4.4 Theorem (Abschlusseigenschaften (1))
Seien A, B reguläre Sprachen. Dann gilt:

1. $A \cap B$ ist regulär.
2. $A \cup B$ ist regulär.
3. AB ist regulär.
4. A^* ist regulär.
5. $\Sigma^* \setminus A$ ist regulär.
6. $A \setminus B$ ist regulär.

2.4.5 Definition (Wort-Homomorphismen)
Eine Abbildung $h : \Sigma^* \to \Sigma'^*$ ist ein *Wort-Homomorphismus*, falls

$$h(\varepsilon) = \varepsilon$$
$$\forall x, y \in \Sigma^* . h(x \cdot y) = h(x) \cdot h(y)$$

2.4.6 Lemma Ein Wort-Homomorphismus $h : \Sigma^* \to \Sigma'^*$ ist bereits vollständig durch sein Σ-Bild definiert, d.h., durch eine Abbildung $f : \Sigma \to \Sigma'^*$.

2.4.7 Theorem (Abschlusseigenschaften (2))
Seien Σ, Σ' zwei Alphabete und $h : \Sigma^* \to \Sigma'^*$ ein Wort-Homomorphismus. Dann gilt:

7. Falls $A \subseteq \Sigma^*$ regulär ist, dann ist auch $h(A)$ regulär.
8. Falls $B \subseteq \Sigma'^*$ regulär ist, dann ist auch $h^{-1}(B)$ regulär.

3 Kontextfreie Sprachen und Kellerautomaten

3.1 Syntaxbäume und Normalformen

Bäume sind Graphen einer besonders regelmäßigen Form. Sie werden im Modul „Diskrete Strukturen“ im Detail eingeführt. Wir benutzen hier eine Repräsentation, die sich besonders für geordnete Bäume eignet. Sie verwendet Wörter über $\mathbb{N}^+$ als direkten Zugriff auf die Knoten. Um zu betonen, dass Knotenwörter zusätzlichen Bedingungen genügen sollen, benutzen wir spitze Klammern anstelle von runden Klammern.

3.1.1 Definition (Knotenmarkierung)
Sei $K \triangleq \{ \langle n_1, \ldots, n_t \rangle \mid t \in \mathbb{N} \text{ und } n_i \in \mathbb{N}^+ \text{ für alle } 1 \leqslant i \leqslant t \}$.
die Menge aller Knotenmarkierungen.
Dabei nennen wir t auch *Tiefe*; die Knotenmarkierung $\langle\rangle$ hat die Tiefe 0.
Seien $x = \langle n_1, \ldots, n_t \rangle$ und $y = \langle m_1, \ldots, m_l \rangle$.
Dann ist $x \cdot y$ gegeben wie in Definition 1.1.2.
Sei $x \leqslant y$ genau dann, wenn $t \leqslant l$ und $n_i = m_i$ für alle $1 \leqslant i \leqslant t$.
Wir schreiben $x < y$, wenn $x \leqslant y$ und $y \neq x$.

3.1.2 Definition (Geordneter Baum)
Ein *geordneter Baum* ist eine endliche Menge $B \subset K$ mit:

1. Wenn $y \in B$ und $x \leqslant y$, dann auch $x \in B$.
2. Wenn $n \in \mathbb{N}^+$ und $x \cdot \langle n{+}1 \rangle \in B$, dann auch $x \cdot \langle n \rangle \in B$.

Die Elemente $x \in B$ heißen *Knoten des Baums*; $\langle\rangle$ heißt *Wurzel*.
Ein $x \in B$ heißt *Blatt*, wenn es kein $y \in B$ gibt, so dass $x < y$.

3.1.3 Bemerkung
Die Adressierung von Knoten in Definition 3.1.2
mit den Markierungen aus Definition 3.1.1
wird gelegentlich auch *universelle Adressierung* genannt.
Tiefensuche in B bezeichnet die Traversierung von B
entlang der lexikographischen Ordnung $\ll_\leqslant$ auf B.
Breitensuche in B bezeichnet die Traversierung von B
entlang der Standardordnung $\ll_\leqslant^S$ auf B.

3.1.4 Definition (Beschrifteter Baum)
Sei L eine beliebige nichtleere Menge.
Sei $B \subset K$ ein Baum und $f : B \to L$ eine Abbildung.
Dann ist das Paar (B, f) ein L-*beschrifteter Baum*.

3.1.5 Definition (Annotierte Ableitung)
Sei $G \triangleq (V, \Sigma, P, S)$ eine Grammatik.
Sei $(\sigma_i)_{i \in [0,n]}$ für $n \in \mathbb{N}$ eine Ableitung in G mit $S = \sigma_0$ und $\sigma_n = w \in \Sigma^*$.
Die *annotierte Ableitung* ersetzt die Elemente $\sigma_i \in (V \cup \Sigma)^*$
durch $\widehat{\sigma_i} \in ((V \times K) \cup (\Sigma \times K))^*$ iterativ wie folgt:

1. Wir schreiben a^x für $(a, x) \in (V \times K) \cup (\Sigma \times K)$.
2. $\widehat{\sigma_0} \triangleq S^{\langle\rangle}$.

3. Sei $\sigma_i \Rightarrow_G \sigma_{i+1}$ wegen

   - $\sigma_i = \alpha_i \cdot \pi \cdot \eta_i$
   - $\sigma_{i+1} = \alpha_i \cdot \pi' \cdot \eta_i$, und
   - $\pi \to_G \pi'$ mit $\pi' = (b_1, \ldots, b_{p_i})$.

   Sei $\widehat{\sigma_i} = \widehat{\alpha_i} \cdot \pi^{x_i} \cdot \widehat{\eta_i}$ bereits berechnet.
   Dann ist $\widehat{\sigma_{i+1}} \triangleq \widehat{\alpha_i} \cdot (b_1^{x_i \cdot \langle 1 \rangle}, \ldots b_{p_i}^{x_i \cdot \langle p_i \rangle}) \cdot \widehat{\eta_i}$

3.1.6 Definition (Syntaxbaum)
Sei $G = (V, \Sigma, P, S)$ eine Typ2-Grammatik.
Sei $w \in L(G)$.
Sei $(\sigma_i)_{i \in [0,n]}$ für $n \in \mathbb{N}$ eine Ableitung in G mit $S = \sigma_0$ und $\sigma_n = w \in \Sigma^*$.
Der *Syntaxbaum* (bzw. *Ableitungsbaum*) dieser Ableitung von w ergibt sich aus der annotierten Ableitung gemäß Definition 3.1.5 als ein $(V \cup \Sigma)$-beschrifteter Baum (B, syn), gegeben durch:

- $\mathrm{syn}(\langle\rangle) \triangleq S$
- $\mathrm{syn}(x_i \cdot \langle j \rangle) \triangleq b_j$ für alle $1 \leqslant j \leqslant p_i$
  für jeden Ableitungsschritt $0 \leqslant i \leqslant n-1$.

Der Baum B ergibt sich nach abgeschlossener Iteration gemäß Definition 3.1.5 als die Menge aller definierten Knotenmarkierungen.

3.1.7 Proposition
Sei $G = (V, \Sigma, P, S)$ eine Typ2-Grammatik.
Sei $w \in L(G)$.
Sei $(\sigma_i)_{i \in [0,n]}$ für $n \in \mathbb{N}$ eine Ableitung in G
mit $S = \sigma_0$ und $\sigma_n = w \in \Sigma^*$.
Sei (B, syn) der Ableitungsbaum von w.
Sei $(x_1, \ldots, x_n)$ die Sequenz der Blätter von B bzgl. der Ordnung $\ll_{\leqslant}$.
Dann gilt $\mathrm{syn}((x_1, \ldots, x_n)) = w$.

3.1.8 Definition
Eine Typ2-Grammatik G heißt *eindeutig*, wenn es zu jedem $w \in L(G)$ genau einen Ableitungsbaum gibt; andernfalls heißt G *mehrdeutig*.

Eine Sprache L heißt *inhärent mehrdeutig*, falls es keine eindeutige Typ2-Grammatik für L gibt.

3.1.9 Bemerkung Die Frage, ob eine Sprache inhärent mehrdeutig ist, ist im Allgemeinen algorithmisch nicht entscheidbar.

3.1.10 Definition (Normalform von Typ-2-Grammatiken)
Sei $G = (V, \Sigma, P, S)$ eine kontextfreie Grammatik.
Sei $\varepsilon \notin L(G)$.

1. G ist in *Chomsky-Normalform (CNF)*,
   falls jede Produktion aus P eine der folgenden Formen hat:

   $$X \to YZ \qquad X \to a$$

   mit $X, Y, Z \in V$ und $a \in \Sigma$.
   G heißt dann auch *CNF-Grammatik*.

2. G ist in *Greibach-Normalform*,
   falls jede Produktion aus P die folgende Form hat:

$$X \to aY_1 \cdots Y_k$$

   mit $k \geqslant 0$, $X, Y_i \in V$ für alle $i \in [1, k]$ und $a \in \Sigma$.

Für den Fall $\varepsilon \in L(G)$ gilt die Regelung aus Definition 1.4.2.

3.1.11 Proposition
Zu jeder kontextfreien Grammatik G mit $\varepsilon \notin L(G)$
gibt es eine kontextfreie Grammatik G'
in Chomsky- (oder auch Greibach-) Normalform
mit $L(G) = L(G')$.

3.2 Cocke-Younger-Kasami-Algorithmus

3.2.1 Proposition
Sei $G = (V, \Sigma, P, S)$ eine CNF-Grammatik.
Sei $w = a_1 a_2 \ldots a_n \in \Sigma^*$.
Seien

- $w_{i,j} \triangleq a_i a_{i+1} \ldots a_{i+j-1}$ mit $1 \leqslant i \leqslant n$ und $1 \leqslant j \leqslant n+1-i$
  das Teilwort von w ab Position i mit Länge j.
- $\mathrm{CYK}_w(i,j) \triangleq \{\, A \in V \mid A \Rightarrow_G^* w_{i,j} \,\}$.

Dann gilt:

1. $\mathrm{CYK}_w(i,1) = \{\, A \in V \mid (A \to a_i) \in P \,\}$
2. $\mathrm{CYK}_w(i,j) = \bigcup_{l=1}^{j-1} \{\, A \in V \mid \exists\, B \in \mathrm{CYK}_w(i,l)\ .\ \exists\, C \in \mathrm{CYK}_w(i+l, j-l)\ .\ (A \to BC) \in P \,\}$
3. $w \in L(G)$ genau dann, wenn $S \in \mathrm{CYK}_w(1,n)$.

3.2.2 Bemerkung Aus obiger Proposition ergibt sich ein Algorithmus, bei dem man für ein gegebenes Wort w zunächst die $\mathrm{CYK}_w(i,1)$ berechnet, danach schrittweise die $\mathrm{CYK}_w(i,j)$ bis hin zu $\mathrm{CYK}_w(1,n)$ ableitet, bis schließlich ein einfacher Elementschaftstest das Ergebnis liefert.

3.3 Nichtdeterministische Kellerautomaten

3.3.1 Definition (Potenzmenge II) Sei A eine beliebige Menge. Dann heißt

$$\mathcal{P}_e(A) \triangleq \{\, B \mid B \subseteq A \ \text{ und } \ \#(B) \in \mathbb{N} \,\}$$

die *Menge aller endlichen Teilmengen von* A.

3.3.2 Bemerkung Für eine unendliche Menge A gilt $\mathcal{P}_e(A) \subset \mathcal{P}(A)$.

3.3.3 Definition (PDA)
Ein *Kellerautomat* (PDA) ist ein 7-Tupel

$$M = (Q, \Sigma, \Gamma, \square, \Delta, q_0, F)$$

wobei

- Q ist eine endliche Menge von *Zuständen*,
- Σ ist eine endliche Symbolmenge, das *Eingabealphabet*,
- Γ ist eine endliche Symbolmenge, das *Kelleralphabet*,
- $\square \in \Gamma$ ist das *Keller-Startsymbol*,
- $\Delta : (Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma) \to \mathcal{P}_e(Q \times \Gamma^*)$ ist die *Übergangsfunktion*,
- $q_0 \in Q$ ist der *Startzustand*,
- $F \subseteq Q$ ist eine Menge von *Endzuständen*.

Δ ist eine Abbildung;
falls $\Delta(q, s, X)$ nicht explizit angegeben ist,
gehen wir implizit von $\Delta(q, s, X) = \emptyset$ aus.

3.3.4 Notation Wir verwenden in der Regel die folgenden Metavariablen:

- $p, q, \ldots \in Q$
- $a, b, \ldots \in \Sigma$ und $u, v \ldots \in \Sigma^*$
- $X, Y, Z \in \Gamma$ und $\alpha, \beta, \gamma \in \Gamma^*$

3.3.5 Notation (Graph eines PDA)
Wie bei DFA benutzen wir auch hier eine graphische Repräsentation:

Falls $\Delta(q, t, X) \ni (q', \alpha)$ für $t \in \Sigma \cup \{\epsilon\}$,
zeichnen wir einen Pfeil mit der Beschriftung $t, X/\alpha$ von q nach q'.

3.3.6 Definition (Berechnungen)
Sei $M = (Q, \Sigma, \Gamma, \square, \Delta, q_0, F)$ ein PDA.

1. Eine *Konfiguration* von M ist ein Tripel $(q, w, \gamma) \in Q \times \Sigma^* \times \Gamma^*$.
2. Die binäre Relation $\vdash_M$ auf Konfigurationen von M ist definiert durch:

$$(q, aw, X\beta) \vdash_M (q', w, \alpha\beta) \qquad \text{falls } (q', \alpha) \in \Delta(q, a, X)$$
$$(q, w, X\beta) \vdash_M (q', w, \alpha\beta) \qquad \text{falls } (q', \alpha) \in \Delta(q, \epsilon, X)$$

   (c, c') mit $c \vdash_M c'$ bezeichnet einen *Berechnungsschritt* von M.
3. Eine *Berechnung von M für das Eingabewort w* ist eine Sequenz von Konfigurationen von M,

$$(q_0, w, \square) \vdash_M (q_1, w_1, \gamma_1) \vdash_M \ldots \vdash_M (q_i, w_i, \gamma_i) \vdash_M \ldots$$

   wobei $(q_0, w, \square)$ *Initialkonfiguration* der Berechnung heißt.

wobei $\vdash_M^*$ den reflexiv-transitiven Abschluss von $\vdash_M$ bezeichnet.
Wir schreiben oft $\vdash$ (und $\vdash^*$) anstelle von $\vdash_M$ (und $\vdash_M^*$).

3.3.7 Lemma Sei $M = (Q, \Sigma, \Gamma, \square, \Delta, q_0, F)$ ein PDA.

1. Wenn $(q, x, \alpha) \vdash^* (q', y, \beta)$,
   dann gilt für alle $w \in \Sigma^*$ und $\gamma \in \Gamma^*$:
   $(q, xw, \alpha\gamma) \vdash^* (q', yw, \beta\gamma)$.
2. Wenn $(q, xw, \alpha) \vdash^* (q', yw, \beta)$,
   dann gilt $(q, x, \alpha) \vdash^* (q', y, \beta)$.

3.3.8 Definition (Akzeptierte Sprache eines PDA)
sei $M = (Q, \Sigma, \Gamma, \square, \Delta, q_0, F)$ ein PDA.

1. Die *durch Endzustände akzeptierte Sprache von* M ist definiert durch:

$$L_{End}(M) \triangleq \{\, w \in \Sigma^* \mid \exists q \in F,\ \alpha \in \Gamma^* . (q_0,\ w,\ \square) \vdash^* (q,\ \varepsilon,\ \alpha) \not\vdash \,\}$$

2. Die *durch leeren Keller akzeptierte Sprache von* M ist definiert durch:

$$L_{Kel}(M) \triangleq \{\, w \in \Sigma^* \mid \exists q \in Q . (q_0,\ w,\ \square) \vdash^* (q,\ \varepsilon,\ \varepsilon) \,\}$$

3.3.9 Proposition Sei Σ ein Alphabet und $A \subseteq \Sigma^*$.
Dann sind die beiden folgenden Aussagen äquivalent:

1. Es existiert ein PDA M mit $A = L_{Kel}(M)$.
2. Es existiert ein PDA M mit $A = L_{End}(M)$.

3.3.10 Proposition Sei Σ ein Alphabet und $A \subseteq \Sigma^*$.
Dann sind die beiden folgenden Aussagen äquivalent:

1. Es existiert ein PDA M mit $A = L_{Kel}(M)$.
2. Es existiert eine Typ2-Grammatik G mit $A = L(G)$.

3.4 Deterministische Kellerautomaten

3.4.1 Definition (DPDA)
Ein *deterministischer Kellerautomat* (DPDA) ist Kellerautomat

$$M = (Q, \Sigma, \Gamma, \square, \Delta, q_0, F)$$

wobei für alle $q \in Q$, $s \in \Sigma$ und $X \in \Gamma$ gilt:

$$\#(\Delta(q, s, X)) \ + \ \#(\Delta(q, \epsilon, X)) \quad \leqslant \quad 1$$

3.4.2 Proposition

1. Für alle $A \in \mathcal{L}_3$ existiert ein DPDA M mit $L_{End}(M) = A$.
2. Es existiert $A \in \mathcal{L}_3$, so dass für alle DPDA M: $L_{Kel}(M) \neq A$.

3.4.3 Definition (Deterministisch kontextfreie Sprachen)
Eine Sprache A heißt *deterministisch kontextfrei*,
wenn ein DPDA existiert mit $L_{End}(M) = A$.

3.4.4 Theorem Sei A eine Sprache.

$$A \text{ regulär} \Rightarrow A \text{ deterministisch kontextfrei} \Rightarrow A \text{ kontextfrei.}$$

3.5 Eigenschaften kontextfreier Sprachen

3.5.1 Lemma (Pumping)
Sei Σ ein Alphabet und $A \subseteq \Sigma^*$.
Sei die Formel **PUMP-CFL**(A) definiert durch:

$$\textbf{PUMP-CFL}(A) \triangleq \exists n{\in}\mathbb{N} \,.\, \forall z{\in}A \,.\, \Big(\text{✪} \to \big(\exists u,v,w,x,y \in \Sigma^* \,.\, \text{❶} \wedge \text{❷} \wedge \text{❸} \wedge \text{❹}\big)\Big)$$

mit: ✪ $\triangleq |z| \geqslant n$ ❶ $\triangleq z = uvwxy$ ❷ $\triangleq |vx| \geqslant 1$ ❸ $\triangleq |vwx| \leqslant n$ ❹ $\triangleq \forall k \in \mathbb{N} \,.\, uv^k wx^k y \in A$

Dann gilt:

1. Wenn A kontextfrei ist, dann gilt **PUMP-CFL**(A).
2. Wenn $\neg$ **PUMP-CFL**(A) gilt, dann ist A nicht kontextfrei.

$$\neg\,\textbf{PUMP-CFL}(A) \equiv \forall n{\in}\mathbb{N} \,.\, \exists z{\in}A \,.\, \Big(\text{✪} \wedge \big(\forall u,v,w,x,y \in \Sigma^* \,.(\text{❶} \wedge \text{❷} \wedge \text{❸}) \to \neg\text{❹}\big)\Big)$$
$$\neg\text{❹} \equiv \exists k \in \mathbb{N} \,.\, uv^k wx^k y \notin A$$

3.5.2 Theorem (Abschlusseigenschaften (1))
Seien A, B kontextfreie Sprachen über Σ.
Dann gilt:

1. $A \cup B$ ist kontextfrei.
2. AB ist kontextfrei.
3. A^* ist kontextfrei.

und, falls $C \subseteq \Sigma^*$ eine reguläre Sprache ist:

4. $A \cap C$ ist kontextfrei.
5. $A \setminus C$ ist kontextfrei.

Dagegen gilt:

6. $A \cap B$ ist nicht notwendigerweise kontextfrei.
7. $\Sigma^* \setminus A$ ist nicht notwendigerweise kontextfrei.
8. $A \setminus B$ ist nicht notwendigerweise kontextfrei.

Seien Σ, Σ' zwei Alphabete
und $h : \Sigma^* \to \Sigma'^*$ ein Wort-Homomorphismus:

9. Wenn $A \subset \Sigma^*$ kontextfrei ist, dann ist auch $h(A)$ kontextfrei.
10. Wenn $B \subset \Sigma'^*$ kontextfrei ist, dann ist auch $h^{-1}(B)$ kontextfrei.

3.5.3 Theorem (Abschlusseigenschaften (2))
Seien A, B deterministisch kontextfreie Sprachen über Σ.
Dann gilt:

1. $\Sigma^* \setminus A$ ist deterministisch kontextfrei.

Dagegen gilt:

2. A^* ist nicht notwendigerweise deterministisch kontextfrei.
3. $A \cup B$ ist nicht notwendigerweise deterministisch kontextfrei.
4. $A \cap B$ ist nicht notwendigerweise deterministisch kontextfrei.
5. $A \setminus B$ ist nicht notwendigerweise deterministisch kontextfrei.