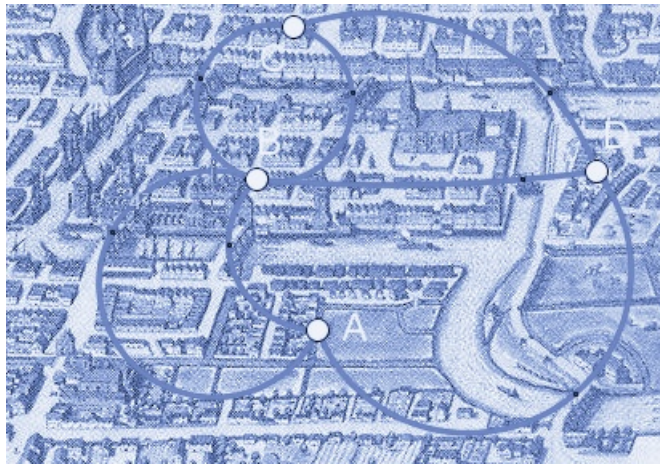


## FORMELSAMMLUNG - MATHEMATIK 3



Zusammenstellung: STL (2008; 19.10.08)

Definition von Injektivität verändert: DOS (2009; 15.09.09)



## CONTENTS

Motivation	3
1. Mengenlehre	4
1.1. Notationen	4
1.2. Darstellung von Mengen	4
1.3. Besondere Mengen	5
1.4. Erste Eigenschaften von Mengen	6
1.5. Mengenoperationen	6
2. Relationen	8
2.1. Kartesisches Produkt	8
2.2. Definition von Relationen	8
2.3. Komposition von Relationen	9
2.4. Eigenschaften von Relationen	10
3. Funktionen and Algorithmen	13
3.1. Definition von Funktionen	13
3.2. Eigenschaften von Funktionen	13
3.3. Rekursive Funktionen	14
3.4. Komplexität von Algorithmen	15
3.5. Zählprinzipien für Mengen	16
4. Graphentheorie	17
4.1. Ungerichtete Graphen	17
4.2. Gerichtete Graphen	22
4.3. Markierte und gewichtete Graphen	25
4.4. Bäume	26
5. Sprachen, Grammatiken und Automaten	28
5.1. Alphabete, Worte und das freie Wortmonoid	28
5.2. Sprachen	29
5.3. Reguläre Sprachen und endliche Automaten	30
5.4. Grammatiken	31
5.5. Endliche Automaten mit Ausgabe	33

## MOTIVATION

Die Untersuchung von mathematischen Problemen wird uns helfen Probleme zu formalisieren, so dass sie von der Maschine eines Rechners verarbeitet werden können. Diese formalen Beschreibungen liefern ein mögliches Modell, um Probleme, insbesondere algorithmisch, zu lösen.

Der Kurs Mathematik 3 ist eine Ergänzung zu den Modulen “Algorithmen und Datenstrukturen“ (Fundamentals 1 / FND1) und “Compilerbau“ (Fundamentals 3 / FND3). Reguläre Ausdrücke werden auch im Bereich “Concurrency“ (FND2 / PRO3) benötigt. Einige Eigenschaften von endlichen Automaten sind im Modul “Modellieren 2“ (MOD 2) ebenfalls von Bedeutung.

Hinweis. Im Folgenden sind im Wesentlichen die Definitionen versammelt. Unterschiedliche Formen der Darstellung, wie von Mengen, Relationen, Funktionen und einfachen Definitionen von Sprachen und Automaten, werden im Einzelnen im Unterricht besprochen; d.h. diese Formelsammlung ersetzt *nicht* das Studium des Buches und der darin enthaltenen zahlreichen Beispiele. Das aktuelle Buch zum Kurs Mathematik 3 ist *S. Lipschutz und M. Lipson: Discrete Mathematics. New York. 1997, 2. Auflage.*

## 1. MENGENLEHRE

In der Mengenlehre werden Mengen und ihre Eigenschaften untersucht.

1.1. **Notationen.** Folgende Notationen für Mengen sollen verwendet werden:

- Eine Menge kann aufgefasst werden als eine Ansammlung von Objekten, die *Elemente einer Menge*.
- Die *Mengen* werden mit Grobuchstaben  $A, B, \dots, X, Y, Z$  und *Elemente in den Mengen* werden mit Kleinbuchstaben  $a, b, \dots, x, y, z$  bezeichnet.
- Wir sagen "ein Element  $p$  ist ein Element in einer Menge  $A$ ", geschrieben  $p \in A$ . (Wir sagen auch " $p$  gehört zu einer Menge  $A$ ").
- Ist  $p$  kein Element der Menge  $A$ , dann schreiben wir  $p \notin A$ .

1.2. **Darstellung von Mengen.** Mengen können auf unterschiedliche Weise spezifiziert werden:

- (1) durch eine (endliche) Aufzählung der Elemente (Wann kann diese Weise der Spezifikation ein Problem sein?); und
- (2) durch Definition von Eigenschaften der Elemente einer Menge.

- Eine Menge  $A$  kann eine *Teilmenge* einer anderen Menge  $B$  sein, geschrieben als  $A \subseteq B$ .
- Eine Menge  $A$  kann eine *echte Teilmenge* einer anderen Menge  $B$  sein, geschrieben als  $A \subset B$ . (Wie unterscheiden sich die Begriffe Teilmenge und echte Teilmenge?)
- Zwei Mengen  $A$  und  $B$  können gleich sein, geschrieben  $A = B$ , dann gilt  $A \subset B$  und  $B \subset A$ .

Bemerkung. Seien  $A$  und  $B$  Mengen. Der Begriff einer *Teilmenge* unterschiedlich (formal) ausgedrückt werden.

$$A \subseteq B = \{x \mid x \in A \Rightarrow x \in B\}$$

Wenn die Operation  $\Rightarrow$  (impliziert) benutzt wird, dann lassen sich Mengen als Träger für eine Boole'sche Algebra interpretieren. Betrachten wir folgende Interpretation des Teilmengenbegriffs:

- (1) Seien  $B = \emptyset$  und  $A = \emptyset$ , dann ist  $\emptyset \subseteq \emptyset$  und damit gilt  $A \subseteq B$  (Die Aussage  $A \subseteq B$  ist *wahr*.)
- (2) Seien  $B = \emptyset$  und  $A \neq \emptyset$ , dann ist  $\emptyset \subseteq A$  und damit gilt  $B \subseteq A$  (Die Aussage  $A \subseteq B$  ist *falsch*.)
- (3) Seien  $B \neq \emptyset$  und  $A = \emptyset$ , dann ist  $\emptyset \subseteq B$  und damit gilt  $A \subseteq B$  (Die Aussage  $A \subseteq B$  ist *wahr*.)
- (4) Seien  $B \neq \emptyset$  und  $A \neq \emptyset$ , dann existiert ein  $x \in A$  und  $x \in B$ , so dass  $A \subseteq B$ . (Die Aussage  $A \subseteq B$  ist *wahr*.)

1.3. **Besondere Mengen.** Für Mengen von Zahlen werden besondere Zeichen eingeführt:

$\mathbb{N}$ : Menge der natürlichen Zahlen;  $\mathbb{N} = \{1, 2, 3, \dots\}$  (ohne Null).

$\mathbb{N}_0$ : Menge der natürlichen Zahlen;  $\mathbb{N}_0 = \{0, 1, 2, \dots\}$  (mit Null).

$\mathbb{Z}$ : Menge der ganzen Zahlen;  $\mathbb{Z} = \{\dots, -2, -1, 0, 1, 2, \dots\}$ .

$\mathbb{Q}$ : Menge der rationalen Zahlen (Menge der endlichen Brüche).

$\mathbb{R}$ : Menge der reellen Zahlen.

$\mathbb{C}$ : Menge der komplexen Zahlen.

Für die Untersuchung von berechenbaren Problemen ist in der Informatik der Unterschied zwischen der Menge der rationalen Zahlen  $\mathbb{Q}$  und der reellen Zahlen  $\mathbb{R}$  wesentlich (warum?).

Um Mengen "erfassen" zu können, machen wir uns das so genannte *Abstraktionsprinzip* zu eigen.

Abstraktionsprinzip. Sei  $U$  eine beliebige Menge und  $P$  eine beliebige Eigenschaft, dann ist  $A$  eine Menge, die genau die Elemente aus  $U$  enthält, die die Eigenschaft  $P$  haben.

Die Menge  $U$  wird auch oftmals als *universale Menge* bezeichnet. (Wozu brauchen wir also die "Festlegung" der universalen Menge? Was ist das "Gegenstück" der universalen Menge? )

*Achtung:* Für eine gegebene Menge  $U$  und eine Eigenschaft  $P$ , mag es Elemente geben, die nicht die Eigenschaft  $P$  haben.

Die leere Menge wird mit  $\emptyset$  bezeichnet. Es gibt nur eine *leere Menge*; d.h. wenn die Mengen  $S$  und  $T$  beide leer sind, dann gilt auch  $S = T$ . Beide Mengen haben dieselben Elemente, nämlich keine!

1.4. **Erste Eigenschaften von Mengen.** Die bisherigen Definitionen können genutzt werden, um einige Eigenschaften von Mengen formal zusammenzufassen:

Theorem.

- Für jede Menge  $A$  gilt:  $\emptyset \subseteq A \subseteq U$ .
- Für jede Menge gilt  $A \subseteq A$  (Idempotenz von Mengen).
- Wenn  $A \subseteq B$  und  $B \subseteq C$  gelten, dann gilt auch  $A \subseteq C$  (Transitivität von Mengen).
- Es gilt  $A = B$  genau dann, wenn  $A \subseteq B$  und  $B \subseteq A$  (Äquivalenz von Mengen).

#### 1.5. Mengenoperationen.

- Die *Vereinigung* zweier Mengen  $A$  und  $B$ , geschrieben  $A \cup B$ , ist die Menge aller Elemente aus der Menge  $A$  und der Menge  $B$ . In Zeichen

$$A \cup B = \{x \mid x \in A \text{ oder } x \in B\}$$

- Der *Durchschnitt* zweier Mengen  $A$  und  $B$ , geschrieben  $A \cap B$ , ist die Menge der Elemente, die beiden Mengen gemeinsam sind. In Zeichen

$$A \cap B = \{x \mid x \in A \text{ und } x \in B\}$$

- Wenn die Mengen  $A$  und  $B$  keine gemeinsamen Elementen haben, dann ist der Durchschnitt  $A \cap B$  leer, geschrieben  $A \cap B = \emptyset$ .
- Ist der Durchschnitt zweier Mengen  $A$  und  $B$  leer ( $A \cap B = \emptyset$ ), dann bezeichnen wir die Mengen  $A$  und  $B$  als *disjunkt*.

- Das *Komplement einer Menge*  $A$ , geschrieben  $A^C$  sind die Elemente, die zwar zu der universalen Menge  $U$  gehören, aber *nicht* Elemente der Menge  $A$  selbst sind (engl. absolute complement). In Zeichen

$$A^C = \{x \mid x \in U \text{ und } x \notin A\}$$

- Die *Differenzmenge*  $A \setminus B$  (oder *Restmenge*) zweier Mengen  $A$  und  $B$  (oder *Komplement der Menge*  $A$  bzgl.  $B$ ) sind die Elemente, die ausschließlich in der Menge  $A$  (aber nicht zu  $B$ ) gehören (engl. relative complement). In Zeichen

$$A \setminus B = \{x \mid x \in A \text{ und } x \notin B\}$$

- Die *symmetrische Differenz* zweier Mengen  $A$  und  $B$  beschreibt, die Menge von Elementen, die jeweils in *einer* der beiden Mengen aber *nicht* im Durchschnitt enthalten sind. In Zeichen

$$\begin{aligned} A \oplus B &= \{x \mid (x \in A) \oplus (x \in B)\} \\ &= (A \cup B) \setminus (A \cap B) \\ &= (A \setminus B) \cup (B \setminus A) \end{aligned}$$

Zum besseren Verständnis der Mengenoperationen ist es empfehlenswert, sich die entsprechenden Venn-Diagramme zu vergegenwärtigen.

Verallgemeinerung von Mengenoperationen. Bisher sind die Mengenoperationen nur für eine binäre (oder zweistellige) Mengenoperation definiert. Diese Mengenoperationen können auf  $n$  verschiedene Mengen  $A_1, A_2, \dots, A_n$  erweitert werden. Es folgen zwei Beispiele.

Seien  $n$  verschiedene Mengen  $A_1, A_2, \dots, A_n$  gegeben. Dann definieren wir ein *verallgemeinertes Produkt* von Mengen durch eine Menge

$$A_1 \cap A_2 \cap \dots \cap A_n,$$

wobei  $1 \leq i \leq n$ . Dieselbe Erweiterung lässt sich für Komplemente von Mengen  $A_1^C, A_2^C, \dots, A_n^C$  definieren durch  $A_1^C \cap A_2^C \cap \dots \cap A_n^C$ .

Potenzmenge. Eine sehr wichtige Menge, die innerhalb der Informatik oft betrachtet wird, ist die so genannte Potenzmenge.

Die Menge mit den Klassen aller Teilmengen für eine gegebene Menge  $S$  wird als Potenzmenge der Menge  $S$  bezeichnet, oft notiert durch  $2^S$  (oder auch  $\mathfrak{P}^S$ ).

## 2. RELATIONEN

In der Mengenlehre werden Mengen und ihre Eigenschaften untersucht, wobei wir nichts über die Anordnungen der Elemente bzgl. dieser Mengen gesagt haben. Die Beziehung zwischen Elementen  $a \in A$  und  $b \in B$  aus den Mengen  $A$  und  $B$ , die wir jetzt als *geordnete Paare*  $(a, b)$  schreiben wollen, werden durch so genannte Relationen festgelegt.

Um diese Relationen und damit Beziehungen zwischen geordneten Paaren ausdrücken zu können, müssen wir *Produktmengen* definieren. Wie im vorausgegangenen Abschnitt beginnen wir mit einem zweistelligen Produkt, dem kartesischen Produkt.

**2.1. Kartesisches Produkt.** Seien  $A$  und  $B$  zwei beliebige Mengen.

- Die Menge der geordneten Paare  $(a, b)$  mit  $a \in A$  und  $b \in B$  nennen wir das *kartesische Produkt*. In Zeichen

$$A \times B = \{(a, b) \mid a \in A \text{ und } b \in B\}$$

- Natürlich kann ein kartesisches Produkt auch über Paaren von Elementen einer Menge definiert werden. Die Menge der geordneten Paare  $(a, a)$  mit  $a \in A$  wird notiert durch  $A \times A$  (oder auch  $A^2$ ).

Die Idee eines Produkts zweier Mengen kann ebenfalls auf eine beliebige endliche Anzahl von Mengen verallgemeinert werden. Für Mengen  $A_1, A_2, \dots, A_n$ , ist die Menge der geordneten  $n$ -Tupel  $(a_1, a_2, \dots, a_n)$  mit  $a_1 \in A_1, a_2 \in A_2, \dots, a_n \in A_n$  ebenfalls ein  $n$ -stelliges Produkt der Mengen  $A_1, A_2, \dots, A_n$ . Dieses Produkt wird geschrieben als  $A_1 \times A_2 \times \dots \times A_n$  oder  $\prod_{i=1}^n A_i$ . Anstatt  $A \times A \times \dots \times A$  können wir, wenn auf die gleiche Menge  $A$  referenziert wird, auch  $A^n$  schreiben.

### 2.2. Definition von Relationen.

**Definition.** Seien  $A$  und  $B$  Mengen. Eine binäre Relation zwischen  $A$  und  $B$  ist eine Teilmenge aus  $A \times B$ .

Sei  $R \subseteq A \times B$  eine Relation zwischen  $A$  und  $B$ . Damit ist die Relation  $R$  eine Menge von geordneten Paaren  $(a, b) \in R$ , wobei das erste Element  $a$  aus der Menge

$A$  ist und das zweite Element aus der Menge  $B$ . Für jedes Paar  $(a, b) \in A \times B$  ist eine der beiden Aussagen in Bezug auf die Relation  $R$  wahr:

- $(a, b) \in R$ ; “ $a$  steht in Relation zu  $b$ “, geschrieben auch als  $aRb$
- $(a, b) \notin R$ ; “ $a$  steht nicht in Relation zu  $b$ “, geschrieben auch als  $a\notin Rb$

Der *Definitionsbereich* (engl. domain  $\mathbb{D}$ ) einer Relation  $R$  ist die Menge aller Elemente an der ersten Position des geordneten Paares  $(a, b) \in R$ , die zu  $R$  gehören. (Wenn  $R \subseteq A \times B$ , dann  $\mathbb{D} \subseteq A$ .)

Der *Wertebereich* (engl. range  $\mathbb{W}$ ) einer Relation  $R$  ist die Menge aller Elemente an der zweiten Position des geordneten Paares  $(a, b) \in R$ , die zu  $R$  gehören. (Wenn  $R \subseteq A \times B$ , dann  $\mathbb{W} \subseteq B$ .)

Relationen lassen sich darstellen durch

- die Auflistung der geordneten Paare in Mengen,
- Tabellen,
- Venn-Diagramme und auch
- gerichtete Graphen.

Inverse Relation. Sei  $R$  eine Relation zwischen einer Menge  $A$  und einer Menge  $B$ . Die *inverse Relation*  $R^{-1}$ , ist die Relation zwischen der Menge  $B$  und der Menge  $A$ , in der die geordneten Paare der Relation  $R$  in umgekehrter Reihenfolge angegeben sind. In Zeichen

$$R^{-1} = \{(b, a) \mid (a, b) \in R\} .$$

Für jede Relation  $R$  gilt  $(R^{-1})^{-1} = R$ .

**2.3. Komposition von Relationen.** Seien  $A$ ,  $B$  und  $C$  Mengen. Sei  $R$  eine Relation von  $A$  nach  $B$  und sei  $S$  eine Relation von  $B$  nach  $C$ . Dann ist  $R \subseteq A \times B$  eine Teilmenge von  $A \times B$  ( $R \subseteq A \times B$ ) und  $S$  ist eine Teilmenge von  $B \times C$  ( $S \subseteq B \times C$ ).  $R$  und  $S$  beschreiben eine Relation von  $A$  nach  $C$ , geschrieben  $R \circ S$  und definiert

durch

$R \circ S = \{(a, c) \mid \text{es gibt ein Element } b \in B, \text{ so dass } (a, b) \in R \text{ und } (b, c) \in S\}$   
 Diese Relation  $R \circ S (\subseteq A \times C)$  heißt *Komposition* der Relationen  $R$  und  $S$ .

*Achtung:* In den meisten Büchern zur “Allgemeinen Algebra“ wird diese Komposition von Relationen durch  $S \circ R$  notiert. Das geschieht in Anlehnung an die Komposition von Funktionen  $g \circ f$ , wobei die Funktionen  $f : A \rightarrow B$  und  $g : B \rightarrow C$  entsprechend definiert sein müssen. (Was ist der wesentliche Unterschied zwischen einer Funktion und einer Relation?)

**2.4. Eigenschaften von Relationen.** Eine Relation  $R$  kann verschiedene Eigenschaften haben, wie sie kann reflexiv, symmetrisch/anti-symmetrisch oder auch transitiv sein. Diese Eigenschaften sollen untersucht und festgelegt werden.

**Reflexivität.** Eine Relation  $R$  auf einer Menge  $A$  ist *reflexiv*, wenn  $aRa$  für jedes Element  $a \in A$  gilt.

Damit ist eine Relation  $R$  auf einer Menge  $A$  *nicht reflexiv*, wenn es ein Element  $a \in A$  gibt, so dass  $(a, a) \notin R$ .

**Symmetrie.** Eine Relation  $R$  auf einer Menge  $A$  ist *symmetrisch*, wenn für jedes Paar  $aRb$  auch  $bRa$  in der Relation  $R$  enthalten ist.

**Antisymmetrie.** Eine Relation  $R$  auf einer Menge  $A$  ist *anti-symmetrisch*, wenn für ein Paar  $(a, b) \in R$ , aber  $(b, a) \notin R$ .

Die Eigenschaft der Anti-Symmetrie lässt sich ebenfalls, wie folgt, formalisieren. Eine Relation  $R$  auf einer Menge  $A$  ist anti-symmetrisch, wenn aus  $aRb$  und  $bRa$  immer  $a = b$  folgt.

Hinweis. Eine Relation kann niemals gleichzeitig symmetrisch und anti-symmetrisch sein; diese beiden Eigenschaften schließen sich aus.

Transitivität. Eine Relation  $R$  auf einer Menge  $A$  ist *transitiv*, für jedes Paar  $aRb$  und für jedes Paar  $bRc$  auch  $aRc$  folgt.

Eine Relation, die sowohl reflexiv, symmetrisch und auch transitiv ist, heisst *Äquivalenzrelation*. Entsprechend lässt sich folgende Definition formulieren:

Äquivalenzrelation. Sei  $A$  eine nicht-leere Menge. Eine Relation  $R$  auf einer Menge  $A$  heißt *Äquivalenzrelation*, wenn sie die folgenden Eigenschaften für Elemente  $a, b, c \in A$  erfüllt:

- (1) Für jedes  $a$  gilt  $aRa$ . (Reflexivität)
- (2) Wenn  $aRb$ , dann auch  $bRa$ . (Symmetrie)
- (3) Wenn  $aRb$  und  $bRc$ , dann auch  $aRc$ . (Transitivität)

Ebenso kann eine Relation  $R'$  reflexiv, anti-symmetrisch und transitiv sein. Eine Relation  $R'$  beschreibt eine *Ordnungsrelation*.

Ordnungsrelation. Sei  $A$  eine Menge und  $\leq$  das Relationssymbol für eine (*Halb-*) *Ordnung* (oder *partielle Ordnung*), dann lässt sich für  $a, b, c \in A$  schreiben

- (1) Für jedes  $a$  gilt  $a \leq a$ . (Reflexivität)
- (2) Wenn  $a \leq b$  und  $b \leq a$ , dann ist impliziert  $a = b$ . (Anti-Symmetrie)
- (3) Wenn  $a \leq b$  und  $b \leq c$ , dann ist auch  $a \leq c$ . (Transitivität)

(Wann ist eine Ordnung *total*?)

Hüllenoperator. Sei  $A$  eine Menge und  $X, Y \subseteq A$ . Eine Abbildung  $\mathcal{H} : \mathcal{P}(A) \rightarrow \mathcal{P}(A)$  heißt Hüllenoperator auf  $A$ , falls für alle Teilmengen  $X, Y$  gilt

- (1)  $X \subseteq \mathcal{H}(X)$  (Extensivität)
- (2)  $X \subseteq Y \Rightarrow \mathcal{H}(X) \subseteq \mathcal{H}(Y)$  (Monotonie)
- (3)  $\mathcal{H}(X) = \mathcal{H}(\mathcal{H}(X))$  (Idempotenz)

## 3. FUNKTIONEN AND ALGORITHMEN

Binäre Relationen ordnen einem Element einer Menge ein oder mehrere Elemente einer anderen Menge zu, wie im vorausgegangenen Abschnitt für binäre Relationen festgelegt. Hingegen Funktionen dienen dazu, jedem Element einer Menge *genau ein* Element einer anderen Menge zu zuordnen.

**3.1. Definition von Funktionen.** Eine *Funktion*  $f : A \rightarrow B$  ist eine Abbildung von einer Menge  $A$  in eine Menge  $B$ , so dass *einem* Element  $a \in A$  *genau ein* Element  $b \in B$  durch  $f(a) = b$  zugeordnet wird.

(engl.  $f : A \rightarrow B$  is a mapping from  $A$  to  $B$ )

**3.2. Eigenschaften von Funktionen.** Eine Funktion kann verschiedene Eigenschaften haben; d.h. sie kann *injektiv*, *surjektiv* und auch *bijektiv* sein.

Sei  $f : A \rightarrow B$  eine Funktion von einer Menge  $A$  in eine Menge  $B$ .

*Injektivität.* Für *alle*  $a \in A$  und  $b \in B$ , wenn  $f(a) = f(b)$  dann  $a = b$ .

(engl.  $f : A \rightarrow B$  is a *one-to-one* mapping from  $A$  to  $B$ )

*Surjektivität.* Für *alle*  $b \in B$ , gibt es ein  $f(a) \in B$  mit  $a \in A$ , so dass  $b = f(a)$ .

(engl.  $f : A \rightarrow B$  is a mapping from  $A$  *onto*  $B$ )

*Bijektivität.* Eine Funktion  $f : A \rightarrow B$ , die sowohl injektiv als auch surjektiv ist, ist auch *bijektiv*.

(engl.  $f : A \rightarrow B$  is a mapping from  $A$  to  $B$  with a *one-to-one correspondence*)

Eine Funktion  $f : A \rightarrow B$  ist *invertierbar*, wenn die inverse Relation auch eine Funktion  $f^{-1}$  von  $B$  nach  $A$  ist ( $f^{-1} : B \rightarrow A$ ).

*Achtung:* Im allgemeinen muss die inverse Relation  $R^{-1}$  keine Funktion sein.

**3.3. Rekursive Funktionen.** Eine auf sich selbst-referenzierende Funktion ist eine *rekursive* Funktion. Dazu muss die Funktion folgende Eigenschaften haben:

- (1) Die Funktion muss einen so genannten Rekursionsanker haben, d.h. einen "Basiswert", für den die Funktion nicht auf sich selbst referenziert.
- (2) Bei jedem sich selbst-referenzierenden Funktionsaufruf muss die noch ausstehende Rekursionstiefe kleiner werden, um schließlich den Rekursionsanker zu erreichen.

Eine rekursive Funktion, die diese beiden Eigenschaften besitzt, bezeichnen wir als *wohl-definiert*. Wohl-definierte Funktionen sind berechenbar.

Sei  $P$  eine Prozedur oder eine rekursive (Berechnungs-)Formel, die angewendet werden kann, um  $f(X)$  zu berechnen, wobei  $f$  die rekursive Funktion ist und  $X$  ist ihr Argument. Die Anzahl der Aufruf von  $P$  bestimmt sich nun wie folgt:

- Der ursprüngliche Aufruf von  $P$  wird mit dem Wert 1 assoziiert.
- Bei jeder Ausführung von  $P$  durch einen rekursiven Aufruf, ist die Anzahl der Aufrufe, die um eins höher ist, als bei dem vorausgegangen Aufruf.
- Die *Rekursionstiefe* von  $f(X)$  ergibt sich aus dem Maximum der Anzahl der Aufrufe von  $P$  während der Berechnung.

Wohl-definierte rekursive Funktionen sind (effektiv) berechenbar, aber sind sie auch effizient berechenbar. Finden wir also einen Algorithmus (ein Berechnungsverfahren), so dass wir die Funktion "schnell" berechnen können? Im Folgenden ist ein Beispiel, die Ackermann Funktion, gegeben, die diese Annahme im allgemeinen widerlegt.

**Ackermann Funktion.** Die *Ackermann Funktion*  $A : \mathbb{N}_0 \times \mathbb{N}_0 \rightarrow \mathbb{N}_0$  ist eine wohl-definierte rekursive Funktion mit zwei Argumenten  $n, m \in \mathbb{N}_0$ . Die Funktion ist definiert durch:

- (1) Wenn  $m = 0$ , dann  $A(m, n) = n + 1$ .
- (2) Wenn  $m \neq 0$ , aber  $n = 0$ , dann  $A(m, n) = A(m - 1, 1)$ .
- (3) Wenn  $m \neq 0$  und  $n \neq 0$ , dann  $A(m, n) = A(m - 1, A(m, n - 1))$ .

(Um das extrem schnelle Wachstum der Funktion zu studieren, ist es empfehlenswert, einmal den Wert von  $A(3, 2)$  zu berechnen.)

**3.4. Komplexität von Algorithmen.** Die *Analyse von Algorithmen* auf ihre Komplexität ist eine wesentliche Aufgabe, um performante Berechnungsverfahren (wenig Verbrauch von Zeit und/oder (Speicher-)Platz) zu garantieren. Komplexitätsbetrachtungen sind auch notwendig, um verschiedene Algorithmen, die das gleiche (mathematische) Problem lösen, in Bezug auf ihre Effizienz zu vergleichen.

*Problem.* Sei  $M$  ein Algorithmus und  $n$  die Größe der (zu verarbeitenden) Eingabedaten. Zeit- und Platzverbrauch eines Algorithmus  $M$  sind die Größen, mit denen sich die Effizienz eines Algorithmus in Abhängigkeit von  $n$  messen lässt.

*Zeit und Platz.* Dazu müssen die wesentlichen Operationen so definiert werden, dass die zu erwartende *Zeit* (zur Ausführung) kleiner oder zumindest proportional anderen Operationen ist (deren (Zeit-)Maß wir kennen). Der *Platz* wird durch Ermitteln des maximalen Speicherverbrauchs bei der Ausführung des Algorithmus abgeschätzt.

*Komplexität eines Algorithmus.* Die Komplexität eines Algorithmus  $M$  ist die Funktion  $f(n)$ , die die Laufzeit und/oder den Platzbedarf des Algorithmus in Abhängigkeit der Größe der Eingabedaten  $n$  nach oben beschränkt (im schlechtesten Fall) angibt.

*Komplexitätsfunktion.* Sei  $M$  ein Algorithmus und  $n$  die Größe der Eingabedaten. Die (zu bestimmende) *Komplexität(-sfunktion)*  $f(n)$  wächst, wenn die Anzahl der Eingaben  $n$  wächst. Es ist üblich, dass Wachstum der Komplexität  $f(n)$  mittels bekannter Funktionen abzuschätzen. Üblicherweise sind solche *Standardfunktionen*, gegenüber denen die Komplexitätsfunktion  $f(n)$  abgeschätzt wird:  $\log_2 n, n, n \log_2 n, n^2, n^3, 2^n$ . Um den tatsächlich die gesuchte Komplexitätsfunktion gegenüber den Standardfunktionen abzuschätzen, verwenden wir die so genannte Groß- $O$ -Notation und diese kann, wie folgt, definiert werden.

*Definition Groß- $O$ -Notation.* Sei eine Funktion  $g : \mathbb{N}_0 \rightarrow \mathbb{R}^+$  gegeben. Dann ist die Menge von Funktionen  $O(g(n)) =: f(n)$ . Damit ist  $f(n)$  selbst eine (ausgewählte) Funktion, deren Funktionswerte ab einem bestimmten Funktionswert  $n_0$ , immer unterhalb von  $g(n)$  liegen, wobei  $c$  eine Konstante ist. Ab diesem Funktionswert  $n_0$  wächst  $f(n)$  also weniger stark als  $g(n)$ . In Zeichen

$$O(g) = \{f : \mathbb{N}_0 \rightarrow \mathbb{R}^+ \mid \text{es existieren } c, n \in \mathbb{R}^+, \text{ so dass } 0 \leq f(n) \leq c * g(n) \text{ für alle } n \geq n_0\}$$

**3.5. Zählprinzipien für Mengen.** Manchmal ist es schwierig (direkt) eine Funktion  $f(n)$  zu finden, die als Komplexitätsmaß  $f : \mathbb{N}_0 \rightarrow \mathbb{R}^+$  geeignet ist, weil die Größe der Eingabedaten  $n$  nur schwierig ermittelt werden kann. Die Bestimmung der Größe der Eingabedaten  $n$  ist besonders schwierig, wenn diese Daten selbst von einer anderen Funktion abhängen. Um dennoch einen Ansatz für eine Abschätzung zu finden, wenden wir uns nochmals den Mengen (auf denen Funktionen definiert sind) zu.

Es kann eine mengentheoretische Interpretation gefunden werden, so dass die Größe der zu erwartenden Eingabedaten  $n$  (ebenfalls) geschätzt werden kann. Dazu bedienen wir uns zwei *Zählprinzipien*: Summenregel-Zählprinzip und Produktregel-Zählprinzip.

Sei  $n(A)$  die Anzahl der Elemente einer Menge  $A$ .

**Summenregel:** Wenn  $A$  und  $B$  disjunkte Mengen sind, dann  
$$n(A \cup B) = n(A) + n(B).$$

**Produktregel:** Sei  $A \times B$  das kartesische Produkt der Mengen  $A$  und  $B$ , dann  
$$n(A \times B) = n(A) * n(B).$$

## 4. GRAPHENTHEORIE

Wie im Abschnitt 2 bei der Betrachtung von Relationen angedeutet, Graphen sind eine der wichtigsten Strukturen in der Informatik. Mit Graphen können insbesondere Abhängigkeiten zwischen Daten beschrieben werden. Graphen bestehen aus Knoten und Kanten. Die Knoten repräsentieren in der Regel die Daten und die Kanten beschreiben die Beziehungen zwischen den Daten. Die Kanten können ungerichtet oder gerichtet sein. Die Eigenschaften von ungerichteten und gerichteten Graphen unterscheiden sich in Vielem, so dass sie in zwei unterschiedlichen Abschnitten behandelt werden.

Eine Unterform gerichteter Graphen sind Bäume. Mit Bäumen lassen sich insbesondere hierarchische Strukturen beschreiben. Hierarchische Strukturen werden in vielen Anwendungen benötigt, um Daten zu nach (vorgegebenen) Kriterien zu ordnen.

## 4.1. Ungerichtete Graphen.

## 4.1.1. Definitionen für ungerichteten Graphen.

Definition eines Graphen. Ein Graph  $G = (V, E)$  besteht aus

- (1) einer Menge  $V = \{v_1, v_2, \dots, v_n\}$  von *Knoten* (vertices,  $n \geq 0$ ) und
- (2) einer Menge  $E \subseteq V \times V$  von *Kanten* (edges).

Definitionen zu Eigenschaften von ungerichteten Graphen.

- Ein Graph  $G$  heißt *knotenendlich*, wenn  $|V| < \infty$  ist; ein Graph heißt *kantenendlich*, wenn  $|E| < \infty$  ist.
- Zwei Knoten heißen *adjazent*, wenn sie durch eine Kante miteinander verbunden sind.
- Die Menge  $N(v)$  von *Nachbarn eines Knoten*  $v \in V$  kann definiert werden durch

$$N(v) = \{w \mid (v, w) \in E \text{ oder } (w, v) \in E\}$$

- Ein ungerichteter Graph ist vollständig durch die Menge der Nachbarn eines jeden Knoten definiert.

- Der *Grad* (*degree*)  $d(v)$  eines Knoten in  $v \in V$  ist gleich der Anzahl Knoten, die  $v$  berühren.
- Die Anzahl der Knoten eines Graphen  $G = (V, E)$  mit einem ungeraden Grad ist ganzzahlig. (Wie kann diese Aussage bewiesen werden?)
- Wenn zwischen zwei Knoten mehrere Kanten verlaufen, nennen wir diese *Parallelkanten*. Sie sind nicht mehr eindeutig durch Angabe von Anfangs- und Endpunkt zu bestimmen.
- Eine *Schlinge* ist eine Kante mit identischem Anfangs- und Endknoten.
- Graphen ohne Schlingen und ohne Parallelkanten werden auch als *einfache Graphen* bezeichnet.
- Ein Graph mit parallelen Kanten aber ohne Schlingen nennen wir *Multi-Graph*.

**Im Folgenden soll der Begriff *Graph* für *einfache Graphen* verwendet werden.**

4.1.2. *Teilgraphen ungerichteter Graphen.* Um Teilgraphen zu definieren, werden wir zuerst weitere Eigenschaften von ungerichteten Graphen festlegen, um dann schließlich auch zwei verschiedene Prinzipien zur Erzeugung von Teilgraphen zu betrachten.

Weitere Begriffe.

- Ein Graph, dessen Knoten allesamt denselben Grad haben, heißt *regulärer Graph*. (Ist der gemeinsame Grad aller Knoten gleich  $k$ , so sprechen wir auch von einem  $k$ -regulären Graphen.)
- Ein *vollständiger Graph* ist ein Graph, in dem *jeder Knoten mit jedem anderen durch eine Kante verbunden* ist.
- Eine *Clique* in einem Graph ist ein maximaler Teilgraph, wobei maximal bedeutet, dass es keine Clique (als Teilgraph) echt enthält.

Sei  $G = (V, E)$  ein vollständiger, ungerichteter Graph ( $|V| = n$ ). Dann ist  $G$  regulär, genauer gesagt  $(n - 1)$ -regulär. Die Zahl der Kanten  $|E|$  von  $G$  bestimmt sich dabei zu

$$|E| = \frac{n(n-1)}{2}$$

Teilgraphen und ihre Erzeugung.

- Einen *Teilgraphen* (oder *Subgraphen*)  $G' = (V', E')$  erhalten wir aus einem Graphen  $G = (V, E)$ , indem wir eine nicht verschwindende Anzahl von Knoten und/oder Kanten eliminieren. Dabei impliziert die Entfernung eines Knoten auch die Entfernung aller ihn berührenden Kanten. Es gilt also  $V' \subseteq V$  und  $E' \subseteq E$ .
- Die Teilgraphen schreiben wir auch als  $G' \subset G$ .
- Ein *Untergraph*  $G' = (V', E')$  eines Graphen  $G = (V, E)$  ist der Graph, der durch die Teilmenge  $V'$  der Knotenmenge induziert wird.  $G'$  hat also die Kantenmenge  $E' = \{(v, w) \mid v, w \in V' \text{ und } (v, w) \in E\}$ .
- Jeder Untergraph ist ein Teilgraph (aber nicht umgekehrt).

4.1.3. *Wege und zusammenhängende Komponenten in ungerichteten Graphen.* Bei Graphen interessiert nicht nur, ob wir von einem Knoten  $v$  direkt über eine Kante zu dem Knoten  $w$  gelangen, sondern ob wir eventuell über Zwischenknoten vom Knoten  $v$  zum Knoten  $w$  kommen können. Dazu führen wir den Begriff eines Weges im Graphen ein.

Weg.

- Ein *Weg*  $p$  in einem Graphen  $G = (V, E)$  ist eine Folge von  $p = w_1, w_2, \dots, w_k$  von Knoten, mit  $(w_i, w_{i+1}) \in E$  für  $1 \leq i < k$ .
- Wir bezeichnen  $w_1$  als *Anfangsknoten des Weges* und  $w_k$  als *Endknoten des Weges*  $p$ .
- Wenn für einen Weg  $p$  der Anfangsknoten  $w_1$  mit dem Endknoten  $w_k$  identisch ist, dann sprechen wir von einem *geschlossenen Weg* (einem *Zyklus* oder *Kreis*).

Eigenschaften von Wegen.

- Wenn auf einem Weg  $p$  jeder Knoten nur einmal vorkommt, bezeichnen wir diesen Weg als *knoteneinfachen Weg*.

- Wenn auf einem Weg  $p$  jede Kante nur einmal durchlaufen wird, dann nennen wir den Weg  $p$  einen *knoteneinfachen Weg*.
- Wenn auf einem Zyklus alle Knoten mit Ausnahme des Anfangsknoten nur einmal vorkommen, sprechen wir von einem *knoteneinfachen Zyklus*.
- Im *ungerichteten Graph* sind Schlingen und Parallelkanten *keine* Zyklen.
  - Wege sind für einfache Graphen definiert, d.h. Parallelkanten gibt es nicht und damit auch keinen entsprechenden Zyklus.
  - Auch Schlingen beschreiben keine Zyklen, weil ein Weg nach obiger Begrifflichkeit mindestens zwei unterschiedliche Knoten umfassen muss.
  - Mit anderen Worten: Ein *Zyklus* hat *in einem ungerichteten Graphen* mindestens *drei Knoten*.
- Der kanteneinfache Zyklus eines Graphen, der alle Kanten umfasst, heißt *Eulerscher Kreis*. Existiert zu einem Graphen  $G$  ein Eulerscher Kreis, dann heißt der Graph  $G$  ein *Eulerscher Graph* (s. a. Titelblatt).
- Der knoteneinfache Zyklus eines Graphen, der alle Knoten berührt, heißt *Hamiltonscher Kreis*. Existiert zu einem Graphen ein Hamiltonscher Kreis, dann heißt der Graph *Hamiltonscher Graph*.
- Die *Länge*  $\|p\|$  des Weges  $p = w_1, w_2, \dots, w_k$  ist gleich der Anzahl der Kanten, die der Weg enthält, also  $\|p\| = k - 1$ .

Das Zusammenfügen von Wegen.

- Zwei Wege  $p_1$  und  $p_2$  eines Graphen heißen *knotendisjunkt*, wenn sie keine gemeinsamen Knoten haben.
- Zwei Wege  $p_1$  und  $p_2$  eines Graphen heißen *kantendisjunkt*, wenn sie keine gemeinsamen Kanten haben.
- Seien  $p_1 = (u_1, u_2, \dots, u_i)$  und  $p_2 = (w_1, w_2, \dots, w_j)$  zwei Wege in einem Graphen  $G$  und es gelte  $u_i = w_1$ , dann können wir einen Gesamtweg  $p_1 \circ p_2$  bilden als  $p_1 \circ p_2 = (u_1, u_2, \dots, u_k, w_2, \dots, w_j)$ .

Zusammenhängende Komponenten.

- Knoten zwischen denen ein Weg existiert, heißen *verbunden* (oder *wechselseitig erreichbar*).

- Die *Verbindung* (oder *wechselseitige Erreichbarkeit*) ist eine Äquivalenzrelation auf den Knoten des Graphen, die eine Aufteilung (Partition) der Knotenmenge  $V$  in disjunkte Teilmengen  $V_1, V_2, \dots, V_k$  erlaubt.
- Zwei Knoten  $u, v \in V$  sind genau dann in derselben Teilmenge  $V_j$ , wenn sie durch einen Weg in  $G = (V, E)$  verbunden sind. Die durch die Partition  $V_1, V_2, \dots, V_k$  induzierten Untergraphen heißen *zusammenhängende Komponenten* des Graphen.
- Für den Fall  $k = 1$ , wenn also der Graph aus einer zusammenhängenden Komponente besteht, d.h. wenn zwischen beliebigen Knotenpaaren des Graphen ein Weg existiert, nennen wir den Graphen selbst *zusammenhängend*.

Nun können wir auch die Existenz von Eulerschen Kreisen charakterisieren.

Ein ungerichteter Graph  $G$  hat genau einen Eulerschen Kreis, wenn gilt

- $G$  ist zusammenhängend und
- jeder Knotengrad von  $G$  ist gerade.

Eine entsprechende Aussage für Hamitonsche Kreise ist ungleich schwieriger. Eine einfache Charakterisierung von Hamitonschen Kreisen liegt nicht vor. Aus der Tatsache erwachsen algorithmische Schwierigkeiten. Während wir effiziente Algorithmen für die Überprüfung, ob ein Graph ein Eulerscher Graph ist, finden können, ist kein effizienter Algorithmus für die Überprüfung von Hamitonschen Graph bekannt.

Eigenschaften zusammenhängender Graphen.

- Ein Knoten  $v$  eines zusammenhängenden Graphen  $G$  heißt *Trennknoten*, wenn bei seiner Entfernung (und bei der Entfernung der in berührenden Kanten) der Graph in mehrere zusammenhängende Komponenten zerfällt.
- Mit anderen Worten:  $v \in V$  ist ein *Trennknoten* des zusammenhängenden Graphen  $G = (V, E)$ , wenn der Untergraph  $G' = (V \setminus \{v\}, E')$  nicht zusammenhängend ist.
- Die *Trennmeng*e eines Graphen enthält alle Trennknoten des Graphen.
- Kanten, deren Elimination eine Aufspaltung des Graphen bewirken, heißen *Schnittkanten*.

## 4.2. Gerichtete Graphen.

4.2.1. *Definitionen für gerichteten Graphen.* Ein gerichteter Graph  $G = (V, E)$  wird ebenfalls durch seine Knotenmenge  $V = \{v_1, v_2, \dots, v_n\}$  ( $n \geq 1$ ) und Kantenmenge  $E \subseteq V \times V$  festgelegt, s.a. "Definitionen für ungerichtete Graphen".

Die wesentlichen Unterschiede ergeben sich bei der Beschreibung der Eigenschaften von gerichteten Graphen.

Eigenschaften von gerichteten Graphen.

- Die *Kantenmenge* enthält eine *Menge von geordneten Paaren*  $(u, v)$ , wobei  $u$  immer der Anfangsknoten ist und  $v$  den Endknoten festlegt.
- Wenn Kanten in ungerichteten Graphen durch Linien dargestellt werden konnten, so sind jetzt die Verbindungen als Pfeile zu zeichnen. Der Anfang eines Pfeils beginnt im Anfangsknoten  $u$  und die Pfeilspitze zeigt auf den Endknoten  $v$  eines Paaren  $(u, v) \in E$ .
- Bei gerichteten Graphen  $G = (V, E)$  können wir zu jedem Knoten  $v \in V$  die Menge der *Vorgängerknoten*  $P(v)$  (predecessor) angeben:
 
$$P(v) = \{w \mid (w, v) \in E\}.$$
- Bei gerichteten Graphen  $G = (V, E)$  können wir zu jedem Knoten  $v \in V$  auch die Menge seiner *Nachfolger*  $S(v)$  (successor) angeben:
 
$$S(v) = \{w \mid (v, w) \in E\}.$$
- Ein gerichteter Graph ist vollständig durch die Nachfolgermenge für jeden Knoten *oder* die Vorgängermenge für jeden Knoten definiert.
- Bei gerichteten Graphen kann die Angabe des Grads eines Knoten  $v$  genauer vorgenommen werden. Wir unterteilen für einen Knoten  $v$ 
  - in *Außengrad*  $d^+(v)$  der die Zahl der Kanten (*outer degree*) angibt, die  $v$  als Anfangsknoten haben, und
  - den *Innengrad*  $d^-(v)$ , der die Zahl der Kanten (*inner degree*) angibt, die  $v$  als Endknoten haben.
- Eine *Schlinge* ist eine Kante mit identischem Anfangs- und Endknoten (ohne Zwischenknoten).

Inverser Graph. Sei  $G = (V, E)$  ein gerichteter Graph. Mit  $G^{-1} = (V, E^{-1})$  bezeichnen wir den zu  $G$  *inversen Graphen*, für den gilt

$$E^{-1} = \{(u, v) \mid (v, u) \in E\}$$

Der Richtungssinn der Kanten von  $G$  wird also gerade umgekehrt.

4.2.2. *Teilgraphen gerichteter Graphen.* Teilgraphen werden für gerichtete Graphen in gleicher Weise erzeugt, wie für ungerichtete Graphen. Das Gleiche gilt für Untergraphen, s. Abschnitt 4.1.2 "Teilgraphen ungerichteter Graphen".

4.2.3. *Wege.* Wie für ungerichtete Graphen lässt sich für gerichtete Graphen der Begriff eines Weges definieren.

Weg. Ein gerichteter Weg  $p$  in einem gerichteten Graphen  $G = (V, E)$  ist eine Folge  $p = w_1, w_2, \dots, w_k$  von Knoten mit  $(w_i, w_{i+1}) \in E$  für  $1 \leq i < k$ , wobei  $w_i$  der Anfangsknoten und  $w_{i+1}$  der Endknoten entsprechend der Richtung der jeweiligen Kante  $(w_i, w_{i+1})$  ist.

Eigenschaften von Wegen in gerichteten Graphen lassen sich in vielen Fällen in gleicher Weise charakterisieren, wie in ungerichteten Graphen. In gerichteten Graphen gibt es also auch: knoteneinfache Wege, kanteneinfache Wege und Zyklen. (Auch hier gilt: Schlingen sind keine Zyklen, weil ein Weg mindestens zwei unterschiedliche Knoten umfassen muss; d.h. ein Zyklus hat in einem gerichteten Graphen mindestens zwei Knoten, da es in gerichteten Graphen keine Parallelkanten gibt.) Die Länge  $\|p\|$  des Weges  $p = w_1, w_2, \dots, w_k$  bleibt gleich der Anzahl Kanten, die der Weg enthält. Wege können auch weiterhin miteinander verbunden werden.

Wegegraph. Die Frage, zwischen welchen Knotenpaaren eines gerichteten Graphen ein Weg existiert, lässt sich Form eines speziellen Graphen, dem so genannten *Wegegraphen* beantworten.

In dem Wegegraphen  $G^+ = (V, E^+)$  zu dem Graph  $G = (V, E)$  existiert zwischen zwei Knoten  $u, v$  genau dann eine Kante, wenn in  $G$  zwischen  $u$  und  $v$  ein Weg

existiert; d.h. also

$$E^+ = \{(u, v) \mid \text{es existiert ein Weg } p = (u, \dots, v) \text{ in } G \text{ und } \|p\| \geq 1\}$$

Zusätzliche Eigenschaften für gerichtete Graphen.

- Ein gerichteter Graph  $G = (V, E)$  heißt genau dann *azyklisch*, wenn es keine Zyklen gibt.
- In jedem azyklischen Graphen  $G$  gibt es mindestens einen Knoten  $v$  mit  $d^+(v) = 0$  und mindestens einen Knoten  $w$  mit  $d^-(w) = 0$ .
- Azyklische Graphen beschreiben partielle Ordnungen: Sei  $G = (V, E)$  ein azyklischer Graph. Sei  $G^+ = (V, E^+)$  der dazugehörige Wegegraph und  $G^* = (V, E^*)$ , mit

$$E^* = E^+ \cup \{(v, v) \mid v \in V\}.$$

Dann kann eine partielle Ordnung  $\leq_G$  folgendermassen definiert werden:

$$v_i \leq_G v_j \iff (v_i, v_j) \in E^*.$$

Sei umgekehrt eine partielle Ordnung  $\leq$  auf der Menge  $V = \{v_1, v_2, \dots, v_k\}$  gegeben. Führen wir dazu die Menge

$$E_{\leq} = \{(u, v) \mid u, v \in V \text{ und } u \leq v \text{ und } u \neq v \text{ und } (u \leq w \leq v, u \neq w \Rightarrow w = v)\}$$

ein, dann ist  $G_{\leq} = (V, E_{\leq})$  ein azyklischer Graph.

4.2.4. *Zusammenhängende Graphen für gerichtete Graphen.* Wir können den Begriff des Zusammenhangs auch für gerichtete Graphen einführen, wobei jetzt zwei unterschiedliche Festlegungen getroffen werden müssen.

Zum einen können wir den Zusammenhang (engl. *connectivity*) auf einem gerichteten Graphen selbst einführen, was bedeutet, dass dazu zwischen beliebigen Knotenpaaren ein gerichteter Weg existiert. Wir sprechen später von *starkem Zusammenhang* (*strongly connected*).

Zum anderen können wir auch den Zusammenhang bezogen auf den unterliegenden ungerichteten Graphen eines gerichteten Graphen betrachten. Den entsprechenden Graphen erhalten wir, wenn alle gerichteten Kanten zu gerichteten Kanten verändert werden. In diesem Fall sprechen wir von einem *schwachen Zusammenhang* des gerichteten Graphen (engl. *weakly connected*).

Folgende Formalisierung dieser Begriffe ist möglich.

Sei  $G = (V, E)$  ein endlicher gerichteter Graph und sei  $G' = (V, E')$  der dazugehörige ungerichtete Graph (d.h. der Graph ohne Kanten mit markierten Richtungen).

- $G$  ist *stark zusammenhängend* (oder zusammenhängend), wenn für jedes Knotenpaar  $(u, v) \in E$  ein Weg von  $u$  nach  $v$  und ein Weg von  $v$  nach  $u$  existiert (d.h. jeder Knoten ist von jedem anderen Knoten der festgelegten Kantenrichtung folgend erreichbar).
- $G$  ist *schwach zusammenhängend*, wenn für jedes Knotenpaar  $(u, v) \in E'$  ein Weg von  $u$  nach  $v$  existiert.

Ein gerichteter Graph  $G$  besitzt genau dann einen Eulerschen Kreis, wenn gilt:

- (1)  $G$  ist schwach zusammenhängend und
- (2) für jeden Knoten  $v$  von  $G$  ist  $d^+(v) = d^-(v)$ .

**4.3. Markierte und gewichtete Graphen.** Wir wollen Knoten- und Kantenmarkierungen von Graphen einführen.

**Knotenmarkierung.** Die *Knotenmarkierung* eines Graphen  $G = (V, E)$  ist eine Abbildung

$$f : V \rightarrow S,$$

wobei  $S$  eine beliebige Wertemenge ist.

- Häufig wird  $S$  als *Farbenmenge* bezeichnet

**Kantenmarkierung.** Die *Kantenmarkierung* eines Graphen  $G = (V, E)$  ist die Abbildung

$$g : V \times V \rightarrow S,$$

wobei  $S$  eine beliebige Wertemenge ist.

- Wir nennen die Abbildung  $g$  auch die *Gewichte* der Kanten.
- Häufig werden die Zahlen als Kantenmarkierungen verwendet und oft als Länge interpretiert.
- Klassische Problemstellungen in kantenmarkierten Graphen sind so genannte Wege- und Flußprobleme.

4.4. **Bäume.** Es gibt eine Reihe von Möglichkeiten Bäume zu definieren. Wir stellen die folgenden zwei Definitionen an den Anfang der Betrachtung, wobei die erste die ungerichtete und die zweite gerichtete Bäume zugrunde legt.

4.4.1. *Definitionen von Bäumen.*

- Ein *ungerichteter Baum* ist ein *ungerichteter, zyklensfreier, zusammenhängender Graph*.
- Ein *gerichteter Baum* ist ein *azyklischer Graph, in dem ein Knoten keinen, alle anderen Knoten einen Vorgänger haben*.

Im Folgenden sollen die betrachteten Bäume gerichtet sein. In der Literatur werden ungerichtete Bäume auch als *Arboreszenzen* bezeichnet.

4.4.2. *Weitere Begriffe zur Beschreibung von gerichteten Bäumen.*

- Der Knoten eines Baumes, der keinen Vorgänger besitzt, heißt *Wurzel* (oder auch *Wurzelknoten*).
- Die Knoten die keinen Nachfolger besitzen, heißen die *Blätter* des Baumes.
- Knoten eines Baumes, die weder Blätter noch Wurzel sind, heißen *innere Knoten*.

Sei  $G = (V, E)$  ein gerichteter Graph. Die folgenden Aussagen sind äquivalent:

- (1)  $G$  ist ein gerichteter Baum.
- (2)  $G$  hat eine Wurzel, von der aus genau ein Weg zu jedem anderen Knoten existiert.
- (3)  $G$  hat eine Wurzel und der  $G$  entsprechende ungerichtete Graph ist ein ungerichteter Baum.
- (4)  $G$  ist schwach zusammenhängend, hat eine Wurzel und jeder Knoten ungleich der Wurzel hat genau einen Vorgänger.

- Die Anzahl Nachfolger eines Knoten heißt *Ordnung des Knotens*.
- Die maximale in einem Baum auftretende Ordnung eines Knotens heißt *Ordnung des Baumes*.
- Ein Baum heißt *vollständig*, wenn zu jedem Knoten entweder 0 oder eine für alle Knoten gleiche Anzahl  $k$  von Nachfolgerknoten existiert. ( $k$  ist dann die Ordnung aller inneren Knoten und des Baumes selbst.)

- Die *Tiefe eines Knotens  $v$*  in einem Baum ist die Anzahl der Kanten eines Weges von der Wurzel bis zum Knoten  $v$ .
- Die *Höhe eines Knotens  $v$*  in einem Baum ist die Anzahl der Kanten auf dem längsten Weg von  $v$  zu einem Blatt.
- Die *Höhe eines Baumes* ist die Höhe der Wurzel.
- Die *Stufe (engl. level) eines Knotens* in einem Baum ist die Höhe des Baumes minus der Tiefe des Knotens.

## 5. SPACHEN, GRAMMATIKEN UND AUTOMATEN

Historisch gesehen ist das Interesse an formalen Sprachen in der Betrachtung der Linguistik natürlicher Sprachen und Programmiersprachen und syntaxorientierte Übersetzung von Programmiersprachen begründet. Formale Sprachen können durch “Grammatiken“ oder entsprechende “Automaten“ beschrieben werden. Während Grammatiken Worte einer Sprache “erzeugen“, können Automaten umgekehrt bei vorgelegtem Wort feststellen, ob dieses zur Sprache gehört. Die (durch Grammatiken oder Automaten darstellbaren) Sprachen können nach verschiedenen Kriterien unterteilt werden. Die am meisten angewendete Sprache sind sicherlich die regulären Sprachen, die insbesondere durch reguläre Ausdrücke spezifiziert werden. Reguläre Ausdrücke werden

- zur einfachen Dateiverarbeitung in Betriebssystemen verwendet,
- beim Ansatz Programmieren per Vertrag in objekt-orientierten Programmiersprachen, um u.a. Invarianten zu formulieren oder
- spezieller zur Spezifikation von Vor- und Nachbedingungen, die auch in Teststümpfen eingesetzt werden und
- zur Spezifikation von so genannten Scannern in Compilern und
- in vielen Anwendungen mehr.

Auch kontextfreie Sprachen besitzen interessante Eigenschaften, wenn ein Compiler zur Übersetzung von Programmiersprachen implementiert werden soll.

### 5.1. Alphabete, Worte und das freie Wortmonoid.

- Sei  $A$  eine nicht-leere endliche Menge von Symbolen, auch *Alphabet* genannt.
- Ein *Wort* (oder eine Zeichenkette)  $w = a_1 \dots a_n$  ( $n \geq 0$ ) über dem Alphabet ist eine endliche Folge; so ist  $n = |w|$  die Länge von  $w$ . Bei  $n = 0$  wird die *leere Zeichenkette* (oder das *leere Wort*)  $\varepsilon$  (oder  $\lambda$ ) notiert.
- $v$  heißt *Teilwort* (oder *Infix*) von  $w$ , wenn zwei möglicherweise leere Zeichenketten  $u, x \in A^*$  existieren, so dass  $w = uvx$ .
- $u$  heißt *Präfix* von  $w$ , wenn es eine möglicherweise leere Zeichenkette  $y \in A^*$  gibt, so dass  $w = uy$  ist.
- $x$  heißt *Suffix* von  $w$ , wenn es eine möglicherweise leere Zeichenkette  $y \in A^*$  gibt, so dass  $w = yx$  ist.

- Sei ein Wort  $u = a_1 a_2 \dots a_n$  über dem Alphabet  $A$ . Jede Folge  $w = a_j a_{j+1} \dots a_k$  ( $1 \leq j \leq k \leq n$ ) ist ein Teilwort von  $u$ .

Mit Operationen auf Worten lässt sich eine mathematische Struktur definieren, das freie Wortmonoid über dem Alphabet  $A$ .

- Seien  $u$  und  $v$  zwei Worte über dem Alphabet  $A$ . Die *Konkatenation* (eng. *concatenation*) von  $u$  und  $v$ , geschrieben  $uv$ , ist das Wort, das entsteht, wenn die Buchstaben von  $u$  gefolgt von  $v$  notiert werden.
- Wie für die Konkatenation von Worten können wir festlegen, dass  $u^2 = uu$ ,  $u^3 = uuu$  und allgemein  $u^{n+1} = u^n u$ , wobei  $u$  ein Wort ist.
- Für die Worte  $u, v, x$  ist  $(uv)x = u(vx)$ . Die Konkatenation ist für Worte über einem Alphabet  $A$  assoziativ. Das leere Wort  $\varepsilon$  ist das *neutrale Element*.
- Sei  $M = A^*$  die Menge aller Worte über dem Alphabet  $A$ , inklusive dem leeren Wort  $\varepsilon$ . Mit  $\varepsilon$  als dem Neutralelement ist  $M$  das *freie (Wort-)Monoid* über  $A$ .

## 5.2. Sprachen.

5.2.1. *Definition einer Sprache.* Eine Sprache (engl. language)  $L$  über einem Alphabet  $A$  ist eine Menge von Worten über  $A$ .  $A^*$  ist definiert als die Menge aller Worte über  $A$ . Dann ist eine Sprache  $L$  eine Teilmenge von  $A^*$  ( $L \subseteq A^*$ ).

5.2.2. *Operationen auf Sprachen.* Seien  $L$  und  $M$  Sprachen über einem Alphabet  $A$ . Dann ist die *Konkatenation von Sprachen*  $L$  und  $M$ , geschrieben durch  $LM$ , die Sprache  $LM$ , die wie folgt definiert ist

$$LM = \{uv \mid u \in L \text{ und } v \in M\}$$

Das ist die Sprache  $LM$ , die die Menge aller Worte beschreibt, die durch Konkatenation von einem Wort aus  $L$  mit einem Wort aus  $M$  entsteht. (Wie kann diese Behauptung bewiesen werden?)

Die Konkatenation von Sprachen ist assoziativ, weil die Konkatenation von Worten assoziativ ist.

Das *Produkt einer Sprache*  $L$  ist definiert als

$$L^0 = \{\varepsilon\}, L^1 = L, L^2 = LL, L^{m+1} = L^m L$$

Die *Kleene-Hüllenoperator*  $L^*$  einer Sprache  $L$  ist definiert durch

$$L^* = L^0 \cup L^1 \cup \dots = \bigcup_{k=0}^{\infty} L^k$$

### 5.3. Reguläre Sprachen und endliche Automaten.

5.3.1. *Reguläre Ausdrücke und reguläre Sprachen.* Sei  $A$  ein nicht-leeres endlichen Alphabet  $A$ . Im Folgenden betrachten wir einen regulären Ausdruck  $r$  über  $A$  und eine Sprache  $L(r)$  über  $A$  assoziiert mit einer regulären Ausdruck  $r$ .

Der reguläre Ausdruck  $r$  und die mit diesem Ausdruck assoziierte Sprache  $L(r)$  ist induktiv definiert, wie folgt.

Definition von regulären Ausdrücken. Jeder der folgende Audrückte über  $A$  ist regulär. Seien  $(, ), *, |$  und  $\varepsilon$  Symbole, die nicht im Alphabet  $A$  enthalten sind.

- (1) Das leere Wort  $\varepsilon$  und das Paar “ $()$ “ (leerer Ausdruck) sind reguläre Ausdrücke.
- (2) Jeder Buchstabe  $a$  aus  $A$  ist ein regulärer Ausdruck.
- (3) Wenn  $r$  ein regulärer Ausdruck ist, dann ist auch  $(r^*)$  ein regulärer Ausdruck (Kleene-Operation für Worte).
- (4) Wenn  $r_1$  und  $r_2$  reguläre Ausdrücke sind, dann ist auch  $(r_1 | r_2)$  ein regulärer Ausdruck (Alternative, Auswahl).
- (5) Wenn  $r_1$  und  $r_2$  reguläre Ausdrücke sind, dann ist auch  $(r_1 r_2)$  ein regulärer Ausdruck (Konkatenation).
- (6) Alle reguläre Ausdrücke sind ausschließlich auf diese Weise definiert.

Definition der mit den regulären Ausdrücken assoziierte Sprache. Die Sprache  $L(r)$  über  $A$  ist definert durch einen regulären Ausdruck  $r$  über  $A$

- (1)  $L(\varepsilon) = \{\varepsilon\}$  und  $L(() ) = \emptyset$ , die leere Menge.
- (2)  $L(a) = \{a\}$ , wobei  $a$  ein Buchstabe aus  $A$  ist.
- (3)  $L(r^*) = (L(r))^*$  [Kleene-Hüllenoperator von  $L(r)$ ]
- (4)  $L(r_1 | r_2) = L(r_1) \cup L(r_2)$  (Vereinigung von Sprachen)
- (5)  $L(r_1 r_2) = L(r_1)L(r_2)$  (Konkatenation von Sprachen)

Klammern werden weggelassen, wenn möglich. Dazu gelten folgende Regeln

- Konkatenation und Vereinigung sind assoziativ.
- Gemäss Konvention bindet  $*$  stärker als die Konkatenation und
- die Konkatenation bindet stärker als  $|$ .

Die Menge der durch reguläre Ausdrücke beschreibbaren Sprachen ist genau die Menge der regulären Sprachen.

### 5.3.2. Endliche Automaten und reguläre Sprachen.

Definition eines (deterministischen) endlichen Automaten. Ein deterministischer endlicher Automat (finite state machine)  $M = (A, S, Y, s_0, F)$  ist durch ein 5-Tupel definiert, wobei

- (1)  $A$  ein endliches *Eingabealphabet* ist;
- (2)  $S$  eine *Menge von Zuständen* ist;
- (3)  $Y \subseteq S$  eine *Menge von Endzuständen* ist;
- (4)  $s_0 \in S$  ein *Startzustand* ist und
- (5)  $F : S \times A \rightarrow S$  ist eine *Transitionsfunktion* ist.

Zu einem gegebenen deterministischen endlichen Automaten  $M = (A, S, Y, s_0, F)$  kann die Transitionsfunktion  $F : S \times A \rightarrow S$  induktiv beschrieben werden

- $F(s, \varepsilon) = z$  und
- $F(s, ax) = F(F(s, a), x)$ ,

wobei  $s \in S, x \in A^*$  und  $a \in A$ .

Sprache  $L(M)$  akzeptiert bei einem endlichen Automaten  $M$ . Sei  $M$  ein (deterministischer) endlicher Automat  $M$  mit Eingabealphabet  $A$  definiert eine Sprache  $L(M)$  durch

$$L(M) = \{w \in A^* \mid F(s_0, w) \in Y\}$$

Satz von Kleene. Eine Sprache  $L$  über einem Alphabet  $A$  ist regulär genau dann, wenn es einen (deterministischen) endlichen Automaten  $M$  gibt, so dass  $L = L(M)$ .

## 5.4. Grammatiken.

5.4.1. *Definition einer Grammatik.* Eine *Chomsky-Grammatik* ist ein 4-Tupel  $G = (V, T, S, P)$ , wobei

- (1)  $V$  ein endliche Menge eines so genannten *Vokabulars* ist;

- (2)  $T$  eine endliche *Menge von Terminalen* (d.h. nicht mehr ersetzbaren Symbolen) ist (die *Menge der Nonterminale*  $N$  (oder Variablen) ist gegeben durch  $N = V \setminus T$ );
- (3)  $S \in N$  ein ausgezeichnetes Startsymbol ist;
- (4)  $P \subseteq V^+ \times V^*$  eine endliche Menge von *Produktionen* (oder *Regeln*) ist.

Eine Produktion ist ein geordnetes Paar  $(\alpha, \beta)$ , üblicherweise geschrieben als  $\alpha \rightarrow \beta$ , wobei  $\alpha$  und  $\beta$  Zeichenketten über  $V$  sind. (Jede Produktion muss mindestens eine nicht-leere Zeichenkette auf der linken Regelseite haben.)

5.4.2. *Chomsky-Hierarchie.* Grammatiken können über die Form der Regelsystem klassifiziert werden. Diese Klassifizierung nennen wir Chomsky-Hierarchie (nach Noam Chomsky, der diese 1956 eingeführt hat).

Sei  $G = (V, T, S, P)$  eine (Chomsky-)Grammatik.

**CH-0 Grammatik:** Eine Chomsky-Grammatik vom Typ-0 hat Regeln der Form  $P \subseteq V^+ \times V^*$ ; s. oben. (Eine solche Grammatik erzeugt (allgemein) eine Chomsky-Sprache.)

**CH-1 Grammatik:** Eine Chomsky-Grammatik vom Typ-1 hat Regeln der Form  $P \subseteq V^+ \times V^*$  und Regeln mit  $\alpha \rightarrow \beta$ , wobei  $|\alpha| \leq |\beta|$  oder  $S \rightarrow \varepsilon$ . (Diese Grammatiken erzeugen so genannte kontext-sensitive Sprachen.)

**CH-2 Grammatik:** Eine Chomsky-Grammatik vom Typ-2 hat Regeln der Form  $P \subseteq N \times V^*$ . (Diese Grammatiken heißen auch kontextfrei und sie erzeugen die kontextfreien Sprachen.)

**CH-3 Grammatik:** Eine Chomsky-Grammatik vom Typ-3 hat Regeln der Form  $P \subseteq (N \times TN) \cup (N \times T) \cup (S \times \{\varepsilon\})$  (Diese Grammatiken erzeugen auch die regulären Sprachen und heißen deswegen auch reguläre Grammatiken.)

Alternativ könnten wir für eine Chomsky-Grammatik vom Typ-3 auch fordern, dass sie ausschließlich Regeln der Form  $P \subseteq (N \times NT) \cup (N \times T) \cup (S \times \{\varepsilon\})$  enthält. (Diese Grammatik heißt auch links-linear, weil nur links ein Nonterminal ersetzt werden kann. Entsprechend sind die in der Hierarchie oben angegeben Grammatiken rechts-linear. Es kann gezeigt werden, dass links-lineare als auch rechts-lineare Grammatiken genau die regulären Sprachen erzeugen.)

Chomsky-Hierarchie.  $CH - 3 \subset CH - 2 \subset CH - 1 \subset CH - 0$ .

Die Grammatiken erzeugen also Sprachen, die echte Teilmengen sind.

**5.5. Endliche Automaten mit Ausgabe.** Ein endlicher Automat mit Ausgabe ist ähnlich dem Konzept eines deterministischen endlichen Automaten. Es wird lediglich eine Ausgabefunktion  $g$  hinzugefügt.

5.5.1. *Definition eines endlichen Automaten mit Ausgabe.* Ein deterministischer *endlicher Automat mit Ausgabe* (engl. finite state machine)  $FSM = (A, S, Z, s_0, f, g)$  ist ein 6-Tupel, wobei

- (1)  $A$  ein endliches *Eingabealphabet* ist;
- (2)  $S$  eine endliche *Menge von Zuständen* ist;
- (3)  $Z$  ein *Ausgabealphabet* ist;
- (4)  $s_0 \in S$  ein *Startzustand* ist;
- (5)  $f : S \times A \rightarrow S$  eine *Transitionsfunktion* ist und
- (6)  $g : S \times A \rightarrow Z$  eine *Ausgabefunktion* ist.

Eine solcher Automat lässt sich durch einen gerichteten Graphen oder durch eine Tabelle darstellen. Bei der Notation der Tabelle ist zu beachten, dass die Transitionsfunktion und die Ausgabefunktion zusammengefasst werden und wir oft  $F : S \times A \rightarrow S \times Z$  notieren, die definiert sind durch  $F(s_i, a_j) = (f(s_i, a_j), (g(s_i, a_j)))$ .

Dieser Automat ist so definiert, dass die Ausgabe bei einem Zustandswechsel, d.h. bei der Ausführung einer Transition über  $S \times A$  erfolgt. Einen solcher Automaten wird in der Literatur auch als *Mealy-Automat* bezeichnet. Analog könnten wir einen Automaten definieren, der in einem Zustand die Ausgabe über  $Z$  liefert. Das wäre dann ein so genannter *Moore-Automat*. Es kann gezeigt werden, dass sich die Ausgabefunktion eines Mealy-Automaten in die eines Moore-Automaten überführen lässt und umgekehrt.