Vorlesungsskript

# Einführung in die Theoretische Informatik

Wintersemester 2016/17

Prof. Dr. Johannes Köbler
Humboldt-Universität zu Berlin
Lehrstuhl Komplexität und Kryptografie

2. November 2016

*Inhaltsverzeichnis*

# Inhaltsverzeichnis

# 1 Einleitung

Rechenmaschinen spielen in der Informatik eine zentrale Rolle. In dieser Vorlesung beschäftigen wir uns mit mathematischen Modellen für Maschinentypen von unterschiedlicher Berechnungskraft. Unter anderem lernen wir das Rechenmodell der Turingmaschine (TM) kennen, mit dem sich alle anderen Rechenmodelle simulieren lassen. Ein weiteres wichtiges Thema der Vorlesung ist die Frage, welche Probleme algorithmisch lösbar sind und wo die Grenzen der Berechenbarkeit verlaufen.

Schließlich untersuchen wir die Komplexität von algorithmischen Problemen, indem wir den benötigten Rechenaufwand möglichst gut nach oben und unten abschätzen. Eine besondere Rolle spielen hierbei die NP-vollständigen Probleme, deren Komplexität bis heute offen ist.

**Themen der Vorlesung**

- Welche Rechenmodelle sind für bestimmte Aufgaben adäquat? (Automatentheorie)
- Welche Probleme sind lösbar? (Berechenbarkeitstheorie)
- Welcher Aufwand ist zur Lösung eines algorithmischen Problems nötig? (Komplexitätstheorie)

In den theoretisch orientierten Folgeveranstaltungen wird es dagegen um folgende Themen gehen.

**Thema der Vorlesung Algorithmen und Datenstrukturen**

- Wie lassen sich praktisch relevante Problemstellungen möglichst effizient lösen? (Algorithmik)

**Thema der Vorlesung Logik in der Informatik**

- Mathematische Grundlagen der Informatik, Beweise führen, Modellierung (Aussagenlogik, Prädikatenlogik)

Der Begriff *Algorithmus* geht auf den persischen Gelehrten Muhammed Al Chwarizmi (8./9. Jhd.) zurück. Der älteste bekannte nicht-triviale Algorithmus ist der nach *Euklid* benannte Algorithmus zur Berechnung des größten gemeinsamen Teilers zweier natürlicher Zahlen (300 v. Chr.). Von einem Algorithmus wird erwartet, dass er jede *Problemeingabe* nach endlich vielen Rechenschritten löst (etwa durch Produktion einer Ausgabe). Eine wichtige Rolle spielen Entscheidungsprobleme, bei denen jede Eingabe nur mit ja oder nein beantwortet wird. Problemeingaben können Zahlen, Formeln, Graphen etc. sein. Diese werden über einem *Eingabealphabet* $\Sigma$ kodiert.

**Definition 1.**

*a) Ein* ***Alphabet*** $\Sigma = \{a_1, \ldots, a_m\}$ *ist eine geordnete Menge von endlich vielen* ***Zeichen****.*

*b) Eine Folge* $x = x_1 \ldots x_n$ *von* $n$ *Zeichen heißt* ***Wort*** *(der* ***Länge*** $|x| = n$*).*

*c) Die Menge aller Wörter über* $\Sigma$ *ist*

$$\Sigma^* = \bigcup_{n \geq 0} \Sigma^n,$$

*wobei* $\Sigma^n = \{x_1 \ldots x_n \mid n \geq 0$ *und* $x_i \in \Sigma$ *für* $i = 1, \ldots, n\}$ *alle Wörter der Länge* $n$ *enthält.*

*d) Das (einzige) Wort der Länge* $n = 0$ *ist das* ***leere Wort****, welches wir mit* $\varepsilon$ *bezeichnen.*

*e) Jede Teilmenge* $L \subseteq \Sigma^*$ *heißt* ***Sprache*** *über dem Alphabet* $\Sigma$*.*

**Beispiel 2.** *Sei* $\Sigma$ *ein Alphabet. Dann sind* $\varnothing, \Sigma^*, \Sigma$ *und* $\{\varepsilon\}$ *Sprachen über* $\Sigma$*. Die Sprache* $\varnothing$ *enthält keine Wörter und heißt* ***leere Sprache****. Die Sprache* $\Sigma^*$ *enthält dagegen alle Wörter über* $\Sigma$*, während die Sprache* $\Sigma$ *alle Wörter über* $\Sigma$ *der Länge 1 enthält. Die Sprache*

*$\{\varepsilon\}$ enthält nur das leere Wort, ist also einelementig. Einelementige Sprachen werden auch als* ***Singletonsprachen*** *bezeichnet.*

Da Sprachen Mengen sind, können wir sie bzgl. Inklusion vergleichen. Zum Beispiel gilt

$$\varnothing \subseteq \{\varepsilon\} \subseteq \Sigma^*.$$

Wir können Sprachen auch vereinigen, schneiden und komplementieren. Seien $A$ und $B$ Sprachen über $\Sigma$. Dann ist

- $A \cap B = \{x \in \Sigma^* \mid x \in A, x \in B\}$ der ***Schnitt*** von $A$ und $B$,
- $A \cup B = \{x \in \Sigma^* \mid x \in A \vee x \in B\}$ die ***Vereinigung*** von $A$ und $B$, und
- $\overline{A} = \{x \in \Sigma^* \mid x \notin A\}$ das ***Komplement*** von $A$.

Neben den Mengenoperationen gibt es auch spezielle Sprachoperationen.

**Definition 3.**

- *Das* ***Produkt*** *(****Verkettung****,* ***Konkatenation****) der Sprachen $A$ und $B$ ist*

  $$AB = \{xy \mid x \in A, y \in B\}.$$

  *Ist $A = \{x\}$ eine Singletonsprache, so schreiben wir für $\{x\}B$ auch einfach $xB$.*
- *Die* ***n-fache Potenz $A^n$*** *einer Sprache $A$ ist induktiv definiert durch*

  $$A^n = \begin{cases} \{\varepsilon\}, & n = 0, \\ A^{n-1}A, & n > 0. \end{cases}$$

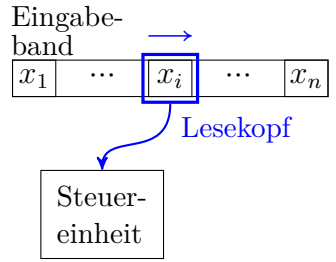
- *Die* ***Sternhülle $A^*$*** *von $A$ ist $A^* = \bigcup_{n \geq 0} A^n$.*
- *Die* ***Plushülle $A^+$*** *von $A$ ist $A^+ = \bigcup_{n \geq 1} A^n = AA^*$.*

# 2 Reguläre Sprachen

Wir betrachten zunächst Einschränkungen des TM-Modells, die vielfältige praktische Anwendungen haben, wie z.B. endliche Automaten (DFA, NFA), Kellerautomaten (PDA, DPDA) etc.

## 2.1 Endliche Automaten

Ein endlicher Automat führt bei einer Eingabe der Länge $n$ nur $n$ Rechenschritte aus. Um die gesamte Eingabe lesen zu können, muss der Automat also in jedem Schritt ein Zeichen der Eingabe verarbeiten.




**Definition 4.** *Ein* ***endlicher Automat*** *(kurz: DFA; deterministic finite automaton) wird durch ein 5-Tupel $M = (Z, \Sigma, \delta, q_0, E)$ beschrieben, wobei*

- *$Z \neq \varnothing$ eine endliche Menge von* ***Zuständen****,*
- *$\Sigma$ das* ***Eingabealphabet****,*
- *$\delta : Z \times \Sigma \to Z$ die* ***Überführungsfunktion****,*
- *$q_0 \in Z$ der* ***Startzustand*** *und*
- *$E \subseteq Z$ die Menge der* ***Endzustände*** *ist.*

*Die von $M$* ***akzeptierte*** *oder* ***erkannte Sprache*** *ist*

$$L(M) = \left\{ x_1 \dots x_n \in \Sigma^* \;\middle|\; \begin{array}{l} \text{es gibt } q_1, \dots, q_{n-1} \in Z, q_n \in E \text{ mit} \\ \delta(q_i, x_{i+1}) = q_{i+1} \text{ für } i = 0, \dots, n-1 \end{array} \right\}.$$

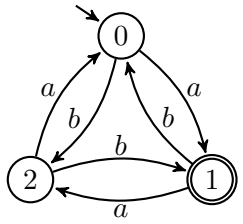
*Eine Zustandsfolge $q_0, q_1, \ldots, q_n$ heißt* ***Rechnung*** *von $M(x_1 \ldots x_n)$, falls $\delta(q_i, x_{i+1}) = q_{i+1}$ für $i = 0, \ldots, n-1$ gilt. Sie heißt* ***akzeptierend****, falls $q_n \in E$ ist, und andernfalls* verwerfend. *Eine von einem DFA akzeptierte Sprache wird als* ***regulär*** *bezeichnet. Die zugehörige Sprachklasse ist*

$$\mathsf{REG} = \{L(M) \mid M \text{ ist ein DFA}\}.$$

**Beispiel 5.** *Betrachte den DFA $M = (Z, \Sigma, \delta, 0, E)$ mit $Z = \{0,1,2\}$, $\Sigma = \{a,b\}$, $E = \{1\}$ und der Überführungsfunktion*

| $\delta$ | 0 | 1 | 2 |
|---|---|---|---|
| $a$ | 1 | 2 | 0 |
| $b$ | 2 | 0 | 1 |

Graphische Darstellung:




Der Startzustand wird meist durch einen Pfeil und Endzustände werden durch einen doppelten Kreis gekennzeichnet.

Bei Eingabe $w_1 = aba$ führt $M$ die akzeptierende Rechnung $0,1,0,1$ durch, d.h. $w_1 \in L(M)$. Dagegen verwirft $M$ das Wort $w_2 = abba$ (verwerfende Rechnung: $0,1,0,2,0$). ◁

Bezeichne $\hat{\delta}(q,x)$ denjenigen Zustand, in dem sich $M$ nach Lesen von $x$ befindet, wenn $M$ im Zustand $q$ gestartet wird. Dann können wir die Funktion

$$\hat{\delta} : Z \times \Sigma^* \to Z$$

induktiv wie folgt definieren. Für $q \in Z$, $x \in \Sigma^*$ und $a \in \Sigma$ sei

$$\begin{aligned} \hat{\delta}(q, \varepsilon) &= q, \\ \hat{\delta}(q, xa) &= \delta(\hat{\delta}(q,x), a). \end{aligned}$$

Die von $M$ erkannte Sprache lässt sich nun auch in der Form

$$L(M) = \{x \in \Sigma^* \mid \hat{\delta}(q_0, x) \in E\}$$



schreiben.

**Behauptung 6.** *Der DFA $M$ aus Beispiel 5 akzeptiert die Sprache*

$$L(M) = \{x \in \Sigma^* \mid \#_a(x) - \#_b(x) \equiv_3 1\},$$

*wobei $\#_a(x)$ die Anzahl der Vorkommen des Zeichens $a$ in $x$ bezeichnet und $i \equiv_m j$ (in Worten: $i$ ist kongruent zu $j$ modulo $m$) bedeutet, dass $i - j$ durch $m$ teilbar ist.*

*Beweis.* Da $M$ nur den Endzustand 1 hat, ist $L(M) = \{x \in \Sigma^* \mid \hat{\delta}(0,x) = 1\}$, d.h. wir müssen folgende Äquivalenz zeigen:

$$\hat{\delta}(0,x) = 1 \Leftrightarrow \#_a(x) - \#_b(x) \equiv_3 1.$$

Hierzu reicht es, die Kongruenz

$$\hat{\delta}(0,x) \equiv_3 \#_a(x) - \#_b(x).$$

zu beweisen, wofür wir Induktion über die Länge $n$ von $x$ benutzen.

***Induktionsanfang*** *($n = 0$)**:*** klar, da $\hat{\delta}(0,\varepsilon) = \#_a(\varepsilon) = \#_b(\varepsilon) = 0$ ist.

***Induktionsschritt*** *($n \rightsquigarrow n+1$)**:*** Sei $x = x_1 \ldots x_{n+1}$ gegeben und sei $i = \hat{\delta}(0, x_1 \ldots x_n)$. Nach IV gilt dann

$$i \equiv_3 \#_a(x_1 \ldots x_n) - \#_b(x_1 \ldots x_n).$$

Wegen $\delta(i,a) \equiv_3 i+1$ und $\delta(i,b) \equiv_3 i-1$ folgt daher

$$\begin{aligned} \delta(i, x_{n+1}) &\equiv_3 i + \#_a(x_{n+1}) - \#_b(x_{n+1}) \\ &\equiv_3 \#_a(x_1 \ldots x_n) - \#_b(x_1 \ldots x_n) + \#_a(x_{n+1}) - \#_b(x_{n+1}) \\ &= \#_a(x) - \#_b(x). \end{aligned}$$

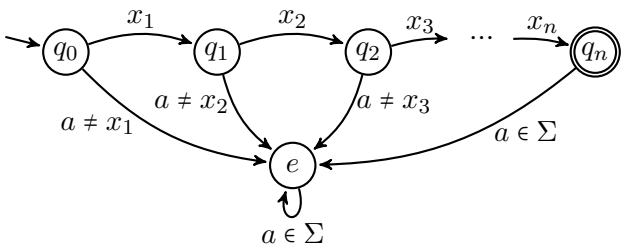
und somit

$$\hat{\delta}(0,x) = \delta(\hat{\delta}(0, x_1 \ldots x_n), x_{n+1}) = \delta(i, x_{n+1}) \equiv_3 \#_a(x) - \#_b(x).$$

■

**Beobachtung 7.** *Alle Singletonsprachen sind regulär.*

*Beweis.* Für jedes Wort $x = x_1 \dots x_n$ existiert ein DFA $M_x$ mit $L(M_x) = \{x\}$:




Formal ist $M_x$ also das Tupel $(Z, \Sigma, \delta, q_0, E)$ mit $Z = \{q_0, \dots, q_n, e\}$, $E = \{q_n\}$ und der Überführungsfunktion

$$\delta(q, a_j) = \begin{cases} q_{i+1}, & q = q_i \text{ für ein } i \text{ mit } 0 \leq i \leq n-1 \text{ und } a_j = x_{i+1} \\ e, & \text{sonst.} \end{cases}$$

■

Als nächstes betrachten wir Abschlusseigenschaften der Sprachklasse REG.

**Definition 8.** *Ein* **k-*stelliger Sprachoperator*** *ist eine Abbildung op, die k Sprachen $L_1, \dots, L_k$ auf eine Sprache $op(L_1, \dots, L_k)$ abbildet.*

**Beispiel 9.** *Der Schnittoperator $\cap$ bildet zwei Sprachen $L_1$ und $L_2$ auf die Sprache $L_1 \cap L_2$ ab.* ◁

**Definition 10.** *Eine Sprachklasse $\mathcal{K}$ heißt unter op* ***abgeschlossen****, wenn gilt:*

$$L_1, \dots, L_k \in \mathcal{K} \Rightarrow op(L_1, \dots, L_k) \in \mathcal{K}.$$

*Der* ***Abschluss*** *von $\mathcal{K}$ unter op ist die bzgl. Inklusion kleinste Sprachklasse $\mathcal{K}'$, die $\mathcal{K}$ enthält und unter op abgeschlossen ist.*



**Beispiel 11.** *Der Abschluss der Singletonsprachen unter $\cap$ besteht aus allen Singletonsprachen und der leeren Sprache.*

*Der Abschluss der Singletonsprachen unter $\cup$ besteht aus allen nichtleeren endlichen Sprachen.* ◁

**Definition 12.** *Für eine Sprachklasse $\mathcal{C}$ bezeichne co-$\mathcal{C}$ die Klasse $\{\bar{L} \mid L \in \mathcal{C}\}$ aller Komplemente von Sprachen in $\mathcal{C}$.*

Es ist leicht zu sehen, dass $\mathcal{C}$ genau dann unter Komplementbildung abgeschlossen ist, wenn *co-*$\mathcal{C} = \mathcal{C}$ ist.

**Beobachtung 13.** *Mit $L_1, L_2 \in$ REG sind auch die Sprachen $\overline{L_1} = \Sigma^* \setminus L_1$, $L_1 \cap L_2$ und $L_1 \cup L_2$ regulär.*

*Beweis.* Sind $M_i = (Z_i, \Sigma, \delta_i, q_0, E_i)$, $i = 1, 2$, DFAs mit $L(M_i) = L_i$, so akzeptiert der DFA

$$\overline{M_1} = (Z_1, \Sigma, \delta_1, q_0, Z_1 \setminus E_1)$$

das Komplement $\overline{L_1}$ von $L_1$. Der Schnitt $L_1 \cap L_2$ von $L_1$ und $L_2$ wird dagegen von dem DFA

$$M = (Z_1 \times Z_2, \Sigma, \delta, (q_0, q_0), E_1 \times E_2)$$

mit

$$\delta((q, p), a) = (\delta_1(q, a), \delta_2(p, a))$$

akzeptiert ($M$ wird auch **Kreuzproduktautomat** genannt). Wegen $L_1 \cup L_2 = \overline{(\overline{L_1} \cap \overline{L_2})}$ ist dann aber auch die Vereinigung von $L_1$ und $L_2$ regulär. (Wie sieht der zugehörige DFA aus?) ■

Aus Beobachtung 13 folgt, dass alle endlichen und alle co-endlichen Sprachen regulär sind. Da die in Beispiel 5 betrachtete Sprache weder endlich noch co-endlich ist, haben wir damit allerdings noch nicht alle regulären Sprachen erfasst.

Es stellt sich die Frage, ob REG neben den mengentheoretischen Operationen Schnitt, Vereinigung und Komplement unter weiteren Operationen wie etwa Produkt oder Sternhülle abgeschlossen ist. Im übernächsten Abschnitt werden wir sehen, dass die Klasse REG als der Abschluss der endlichen Sprachen unter Vereinigung, Produkt und Sternhülle charakterisierbar ist.

Beim Versuch, einen endlichen Automaten für das Produkt $L_1L_2$ zweier regulärer Sprachen zu konstruieren, stößt man auf die Schwierigkeit, den richtigen Zeitpunkt für den Übergang von (der Simulation von) $M_1$ zu $M_2$ zu finden. Unter Verwendung eines nichtdeterministischen Automaten lässt sich dieses Problem jedoch leicht beheben, da dieser den richtigen Zeitpunkt „erraten“ kann.

Im nächsten Abschnitt werden wir nachweisen, dass auch nichtdeterministische endliche Automaten nur reguläre Sprachen erkennen können.

## 2.2 Nichtdeterministische endliche Automaten

**Definition 14.** *Ein **nichtdeterministischer endlicher Automat** (kurz: NFA; nondeterministic finite automaton) $N = (Z, \Sigma, \Delta, Q_0, E)$ ist ähnlich aufgebaut wie ein DFA, nur dass er mehrere Startzustände (zusammengefasst in der Menge $Q_0 \subseteq Z$) haben kann und seine Überführungsfunktion die Form*

$$\Delta : Z \times \Sigma \to \mathcal{P}(Z)$$

*hat. Hierbei bezeichnet $\mathcal{P}(Z)$ die **Potenzmenge** (also die Menge aller Teilmengen) von $Z$. Diese wird auch oft mit $2^Z$ bezeichnet. Die von $N$ akzeptierte Sprache ist*

$$L(N) = \left\{ x_1 \dots x_n \in \Sigma^* \;\middle|\; \begin{array}{c} \exists\, q_0 \in Q_0, q_1, \dots, q_{n-1} \in Z, q_n \in E: \\ q_{i+1} \in \Delta(q_i, x_{i+1}) \text{ für } i = 0, \dots, n-1 \end{array} \right\}.$$

*Eine Zustandsfolge $q_0, q_1, \dots, q_n$ heißt **Rechnung** von $N(x_1 \dots x_n)$, falls $q_{i+1} \in \Delta(q_i, x_{i+1})$ für $i = 0, \dots, n-1$ gilt.*

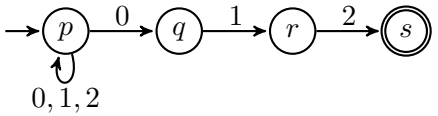
Ein NFA $N$ kann bei einer Eingabe $x$ also nicht nur eine, sondern mehrere verschiedene Rechnungen parallel ausführen. Ein Wort $x$ gehört genau dann zu $L(N)$, wenn $N(x)$ mindestens eine akzeptierende Rechnung hat.

Im Gegensatz zu einem DFA, dessen Überführungsfunktion auf der gesamten Menge $Z \times \Sigma$ definiert ist, kann ein NFA „stecken bleiben“. Das ist dann der Fall, wenn er in einen Zustand $q$ gelangt, in dem das nächste Eingabezeichen $x_i$ wegen $\Delta(q, x_i) = \varnothing$ nicht gelesen werden kann.

**Beispiel 15.** *Betrachte den NFA $N = (Z, \Sigma, \Delta, Q_0, E)$ mit Zustandsmenge $Z = \{p, q, r, s\}$, Eingabealphabet $\Sigma = \{0, 1, 2\}$, Start- und Endzustandsmenge $Q_0 = \{p\}$ und $E = \{s\}$ sowie der Überführungsfunktion*

| $\Delta$ | $p$ | $q$ | $r$ | $s$ |
|---|---|---|---|---|
| 0 | $\{p, q\}$ | $\varnothing$ | $\varnothing$ | $\varnothing$ |
| 1 | $\{p\}$ | $\{r\}$ | $\varnothing$ | $\varnothing$ |
| 2 | $\{p\}$ | $\varnothing$ | $\{s\}$ | $\varnothing$ |

*Graphische Darstellung:*




*Offensichtlich akzeptiert $N$ die Sprache $L(N) = \{x012 \mid x \in \Sigma^*\}$ aller Wörter, die mit dem Suffix 012 enden.* ◁

**Beobachtung 16.** *Sind $N_i = (Z_i, \Sigma, \Delta_i, Q_i, E_i)$ $(i = 1, 2)$ NFAs, so werden auch die Sprachen $L(N_1)L(N_2)$ und $L(N_1)^*$ von einem NFA erkannt.*

*Beweis.* Sei $L_i = L(N_i)$. Wir können $Z_1 \cap Z_2 = \varnothing$ annehmen. Dann akzeptiert der NFA

$$N = (Z_1 \cup Z_2, \Sigma, \Delta_3, Q_1, E)$$

mit

$$\Delta_3(p,a) = \begin{cases} \Delta_1(p,a), & p \in Z_1 \setminus E_1, \\ \Delta_1(p,a) \cup \bigcup_{q \in Q_2} \Delta_2(q,a), & p \in E_1, \\ \Delta_2(p,a), & \text{sonst} \end{cases}$$

und

$$E = \begin{cases} E_2, & Q_2 \cap E_2 = \varnothing \\ E_1 \cup E_2, & \text{sonst} \end{cases}$$

die Sprache $L_1 L_2$.

$L_1 L_2 \subseteq L(N)$*:* Seien $x = x_1 \cdots x_k \in L_1, y = y_1 \cdots y_l \in L_2$ und seien $q_0, \ldots, q_k$ und $p_0, \ldots, p_l$ akzeptierende Rechnungen von $N_1(x)$ und $N_2(y)$. Dann ist $q_0, \ldots, q_k, p_1, \ldots, p_l$ eine akz. Rechnung von $N(xy)$, da $q_0 \in Q_1$ und $p_l \in E_2$ ist, und

- im Fall $l \geq 1$ wegen $q_k \in E_1$, $p_0 \in Q_2$ und $p_1 \in \Delta_2(p_0, y_1)$ zudem $p_1 \in \Delta(q_k, y_1)$ und
- im Fall $l = 0$ wegen $q_k \in E_1$ und $p_l \in Q_2 \cap E_2$ zudem $q_k \in E$ ist.

$L(N) \subseteq L_1 L_2$*:* Sei $x = x_1 \cdots x_n \in L(N)$ und sei $q_0, \ldots, q_n$ eine akz. Rechnung von $N(x)$. Dann gilt $q_0 \in Q_1$, $q_n \in E$, $q_0, \ldots, q_i \in Z_1$ und $q_{i+1}, \ldots, q_n \in Z_2$ für ein $i \leq n$. Wir zeigen, dass ein $q \in Q_2$ existiert, so dass $q_0, \ldots, q_i$ eine akz. Rechnung von $N_1(x_1 \cdots x_i)$ und $q, q_{i+1}, \ldots, q_n$ eine akz. Rechnung von $N_2(x_{i+1} \cdots x_n)$ ist.

- Im Fall $i < n$ impliziert der Übergang $q_{i+1} \in \Delta(q_i, x_{i+1})$, dass $q_i \in E_1$ und $q_{i+1} \in \Delta_2(q, x_{i+1})$ für ein $q \in Q_2$ ist.
- Im Fall $i = n$ ist $q_n \in E_1$ und $Q_2 \cap E_2 \neq \varnothing$ (d.h. $\varepsilon \in L_2$).

Ganz ähnlich lässt sich zeigen, dass der NFA

$$N^* = (Z_1 \cup \{q_{neu}\}, \Sigma, \Delta_4, Q_1 \cup \{q_{neu}\}, E_1 \cup \{q_{neu}\})$$

mit



$$\Delta_4(p,a) = \begin{cases} \Delta_1(p,a), & p \in Z_1 \setminus E_1, \\ \Delta_1(p,a) \cup \bigcup_{q \in Q_1} \Delta_1(q,a), & p \in E_1, \\ \varnothing, & \text{sonst} \end{cases}$$

die Sprache $L_1^*$ akzeptiert. ∎

**Satz 17** (Rabin und Scott)**.**
$\mathsf{REG} = \{L(N) \mid N \textit{ ist ein NFA}\}$.

*Beweis.* Die Inklusion von links nach rechts ist klar, da jeder DFA auch als NFA aufgefasst werden kann. Für die Gegenrichtung konstruieren wir zu einem NFA $N = (Z, \Sigma, \Delta, Q_0, E)$ einen DFA $M = (\mathcal{P}(Z), \Sigma, \delta, Q_0, E')$ mit $L(M) = L(N)$. Wir definieren die Überführungsfunktion $\delta : \mathcal{P}(Z) \times \Sigma \to \mathcal{P}(Z)$ von $M$ mittels

$$\delta(Q,a) = \bigcup_{q \in Q} \Delta(q,a).$$

Die Menge $\delta(Q,a)$ enthält also alle Zustände, in die $N$ gelangen kann, wenn $N$ ausgehend von einem beliebigen Zustand $q \in Q$ das Zeichen $a$ liest. Intuitiv bedeutet dies, dass der DFA $M$ den NFA $N$ simuliert, indem $M$ in seinem aktuellen Zustand $Q$ die Information speichert, in welchen Zuständen sich $N$ momentan befinden könnte. Für die Erweiterung $\hat{\delta} : \mathcal{P}(Z) \times \Sigma^* \to \mathcal{P}(Z)$ von $\delta$ (siehe Seite 3) können wir nun folgende Behauptung zeigen.

**Behauptung.** $\hat{\delta}(Q_0, x)$ enthält alle Zustände, die $N$ ausgehend von einem Startzustand nach Lesen von $x$ erreichen kann.

Wir beweisen die Behauptung induktiv über die Länge $n$ von $x$.

***Induktionsanfang*** *($n = 0$)**:*** klar, da $\hat{\delta}(Q_0, \varepsilon) = Q_0$ ist.

***Induktionsschritt*** *($n - 1 \rightsquigarrow n$)**:*** Sei $x = x_1 \ldots x_n$ gegeben. Nach Induktionsvoraussetzung enthält

$$Q_{n-1} = \hat{\delta}(Q_0, x_1 \ldots x_{n-1})$$

alle Zustände, die $N(x)$ in genau $n-1$ Schritten erreichen kann. Wegen

$$\hat{\delta}(Q_0, x) = \delta(Q_{n-1}, x_n) = \bigcup_{q \in Q_{n-1}} \Delta(q, x_n)$$

enthält dann aber $\hat{\delta}(Q_0, x)$ alle Zustände, die $N(x)$ in genau $n$ Schritten erreichen kann.

Deklarieren wir nun diejenigen Teilmengen $Q \subseteq Z$, die mindestens einen Endzustand von $N$ enthalten, als Endzustände des **Potenzmengenautomaten** $M$, d.h.

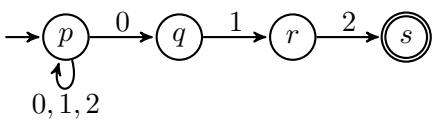
$$E' = \{Q \subseteq Z \mid Q \cap E \neq \varnothing\},$$

so folgt für alle Wörter $x \in \Sigma^*$:

$$\begin{aligned} x \in L(N) &\Leftrightarrow N(x) \text{ kann in genau } |x| \text{ Schritten einen Endzustand erreichen} \\ &\Leftrightarrow \hat{\delta}(Q_0, x) \cap E \neq \varnothing \\ &\Leftrightarrow \hat{\delta}(Q_0, x) \in E' \\ &\Leftrightarrow x \in L(M). \end{aligned}$$

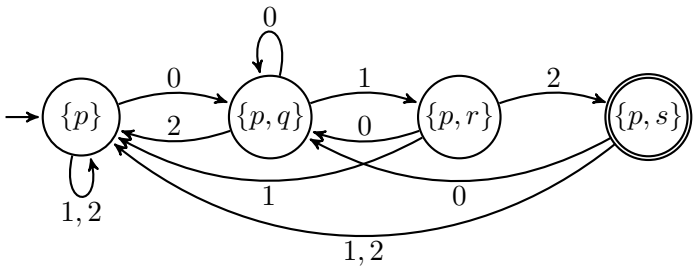
■

**Beispiel 18.** *Für den NFA $N = (Z, \Sigma, \Delta, Q_0, E)$ aus Beispiel 15*




*ergibt die Konstruktion des vorigen Satzes den folgenden DFA $M$ (nach Entfernen aller vom Startzustand $Q_0 = \{p\}$ aus nicht erreichbaren Zustände):*

| $\delta$ | 0 | 1 | 2 |
|---|---|---|---|
| $Q_0 = \{p\}$ | $\{p, q\}$ | $\{p\}$ | $\{p\}$ |
| $Q_1 = \{p, q\}$ | $\{p, q\}$ | $\{p, r\}$ | $\{p\}$ |
| $Q_2 = \{p, r\}$ | $\{p, q\}$ | $\{p\}$ | $\{p, s\}$ |
| $Q_3 = \{p, s\}$ | $\{p, q\}$ | $\{p\}$ | $\{p\}$ |




◁

Im obigen Beispiel wurden für die Konstruktion des DFA $M$ aus dem NFA $N$ nur 4 der insgesamt $2^{\|Z\|} = 16$ Zustände benötigt, da die übrigen 12 Zustände in $\mathcal{P}(Z)$ nicht vom Startzustand $Q_0 = \{p\}$ aus erreichbar sind. Es gibt jedoch Beispiele, bei denen alle $2^{\|Z\|}$ Zustände in $\mathcal{P}(Z)$ für die Konstruktion des Potenzmengenautomaten benötigt werden (siehe Übungen).

**Korollar 19.** *Die Klasse* REG *der regulären Sprachen ist unter folgenden Operationen abgeschlossen:*

- *Komplement,*
- *Schnitt,*
- *Vereinigung,*
- *Produkt,*
- *Sternhülle.*

## 2.3 Reguläre Ausdrücke

Wir haben uns im letzten Abschnitt davon überzeugt, dass auch NFAs nur reguläre Sprachen erkennen können:

$$\mathsf{REG} = \{L(M) \mid M \text{ ist ein DFA}\} = \{L(N) \mid N \text{ ist ein NFA}\}.$$

In diesem Abschnitt werden wir eine weitere Charakterisierung der regulären Sprachen kennenlernen:

> REG ist die Klasse aller Sprachen, die sich mittels der Operationen Vereinigung, Schnitt, Komplement, Produkt und Sternhülle aus der leeren Menge und den Singletonsprachen bilden lassen.

Tatsächlich kann hierbei sogar auf die Schnitt- und Komplementbildung verzichtet werden.

**Definition 20.** *Die Menge der* ***regulären Ausdrücke*** $\gamma$ *(über einem Alphabet* $\Sigma$*) und die durch* $\gamma$ *dargestellte Sprache* $L(\gamma)$ *sind induktiv wie folgt definiert. Die Symbole* $\varnothing$, $\epsilon$ *und* $a$ $(a \in \Sigma)$ *sind reguläre Ausdrücke, die*

- *die leere Sprache* $L(\varnothing) = \varnothing$,
- *die Sprache* $L(\epsilon) = \{\varepsilon\}$ *und*
- *für jedes Zeichen* $a \in \Sigma$ *die Sprache* $L(a) = \{a\}$

*beschreiben. Sind* $\alpha$ *und* $\beta$ *reguläre Ausdrücke, die die Sprachen* $L(\alpha)$ *und* $L(\beta)$ *beschreiben, so sind auch* $\alpha\beta$, $(\alpha|\beta)$ *und* $(\alpha)^*$ *reguläre Ausdrücke, die die Sprachen*

- $L(\alpha\beta) = L(\alpha)L(\beta)$,
- $L(\alpha|\beta) = L(\alpha) \cup L(\beta)$ *und*
- $L((\alpha)^*) = L(\alpha)^*$

*beschreiben.*

**Bemerkung 21.**

- *Um Klammern zu sparen, definieren wir folgende* ***Präzedenzordnung****: Der Sternoperator* $^*$ *bindet stärker als der Produktoperator und dieser wiederum stärker als der Vereinigungsoperator. Für* $((a|b(c)^*)|d)$ *können wir also kurz* $a|bc^*|d$ *schreiben.*
- *Da der reguläre Ausdruck* $\gamma\gamma^*$ *die Sprache* $L(\gamma)^+$ *beschreibt, verwenden wir* $\gamma^+$ *als Abkürzung für den Ausdruck* $\gamma\gamma^*$.

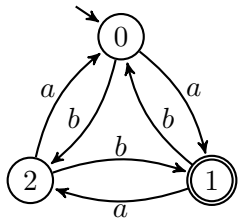


**Beispiel 22.** *Die regulären Ausdrücke* $\epsilon^*$, $\varnothing^*$, $(0|1)^*00$ *und* $\epsilon 0|\varnothing 1^*$ *beschreiben folgende Sprachen:*

| $\gamma$ | $\epsilon^*$ | $\varnothing^*$ | $(0\|1)^*00$ | $\epsilon 0\|\varnothing 1^*$ |
|---|---|---|---|---|
| $L(\gamma)$ | $\{\varepsilon\}^* = \{\varepsilon\}$ | $\varnothing^* = \{\varepsilon\}$ | $\{x00 \mid x \in \{0,1\}^*\}$ | $\{0\}$ |

◁

**Beispiel 23.** *Betrachte nebenstehenden DFA* $M$*. Um für die von* $M$ *erkannte Sprache*




$$L(M) = \{x \in \{a,b\}^* \mid \#_a(x) - \#_b(x) \equiv_3 1\}$$

*einen regulären Ausdruck zu finden, betrachten wir zunächst die Sprache* $L_{0,0}$ *aller Wörter* $x$*, die den DFA* $M$ *ausgehend vom Zustand* 0 *in den Zustand* 0 *überführen. Weiter sei* $L_{0,0}^{\neq 0}$ *die Sprache aller solchen Wörter* $w \neq \varepsilon$*, die zwischendurch nicht den Zustand 0 besuchen. Dann setzt sich jedes* $x \in L_{0,0}$ *aus beliebig vielen Teilwörtern* $w_1, \ldots, w_k \in L_{0,0}^{\neq 0}$ *zusammen, d.h.* $L_{0,0} = (L_{0,0}^{\neq 0})^*$.

*Jedes* $w \in L_{0,0}^{\neq 0}$ *beginnt entweder mit einem* $a$ *(Übergang von* 0 *nach* 1*) oder mit einem* $b$ *(Übergang von* 0 *nach* 2*). Im ersten Fall folgt eine beliebige Anzahl von Teilwörtern* $ab$ *(Wechsel zwischen* 1 *und* 2*), an die sich entweder das Suffix* $aa$ *(Rückkehr von* 1 *nach* 0 *über* 2*) oder das Suffix* $b$ *(direkte Rückkehr von* 1 *nach* 0*) anschließt. Analog folgt im zweiten Fall eine beliebige Anzahl von Teilwörtern* $ba$ *(Wechsel zwischen* 2 *und* 1*), an die sich entweder das Suffix* $a$ *(direkte Rückkehr von* 2 *nach* 0*) oder das Suffix* $bb$ *(Rückkehr von* 2 *nach* 0 *über* 1*) anschließt. Daher lässt sich* $L_{0,0}^{\neq 0}$ *durch den regulären Ausdruck*

$$\gamma_{0,0}^{\neq 0} = a(ab)^*(aa|b) \mid b(ba)^*(a|bb) \mid \epsilon$$

*beschreiben.* Eine ähnliche Überlegung zeigt, dass sich die die Sprache $L_{0,1}^{\neq 0}$ aller Wörter, die $M$ ausgehend von 0 in den Zustand 1 überführen, ohne dass zwischendurch der Zustand 0 nochmals besucht

wird, durch den regulären Ausdruck $\gamma_{0,1}^{\neq 0} = (a|bb)(ab)^*$ beschreibbar ist. Somit erhalten wir für $L(M)$ den regulären Ausdruck

$$\gamma_{0,1} = (a(ab)^*(aa|b) \mid b(ba)^*(a|bb))^*(a|bb)(ab)^*.$$

◁

**Satz 24.** $\{L(\gamma) \mid \gamma$ *ist ein regulärer Ausdruck*$\} = \mathsf{REG}$.

*Beweis.* Die Inklusion von rechts nach links ist klar, da die Basisausdrücke $\varnothing$, $\epsilon$ und $a$, $a \in \Sigma^*$, nur reguläre Sprachen beschreiben und die Sprachklasse REG unter Produkt, Vereinigung und Sternhülle abgeschlossen ist (siehe Beobachtungen 13 und 16).

Für die Gegenrichtung konstruieren wir zu einem DFA $M$ einen regulären Ausdruck $\gamma$ mit $L(\gamma) = L(M)$. Sei also $M = (Z, \Sigma, \delta, q_0, E)$ ein DFA, wobei wir annehmen können, dass $Z = \{1, \ldots, m\}$ und $q_0 = 1$ ist. Dann lässt sich $L(M)$ als Vereinigung

$$L(M) = \bigcup_{q \in E} L_{1,q}$$

von Sprachen der Form

$$L_{p,q} = \{x \in \Sigma^* \mid \hat{\delta}(p, x) = q\}$$

darstellen. Folglich reicht es zu zeigen, dass die Sprachen $L_{p,q}$ durch reguläre Ausdrücke beschreibbar sind. Hierzu betrachten wir die Sprachen

$$L_{p,q}^r = \left\{ x_1 \ldots x_n \in \Sigma^* \;\middle|\; \begin{array}{c} \hat{\delta}(p, x_1 \ldots x_n) = q \text{ und für} \\ i = 1, \ldots, n-1 \text{ gilt } \hat{\delta}(p, x_1 \ldots x_i) \leq r \end{array} \right\}.$$

Wegen $L_{p,q} = L_{p,q}^m$ reicht es, reguläre Ausdrücke $\gamma_{p,q}^r$ für die Sprachen $L_{p,q}^r$ anzugeben. Im Fall $r = 0$ enthält

$$L_{p,q}^0 = \begin{cases} \{a \in \Sigma \mid \delta(p, a) = q\} \cup \{\varepsilon\}, & p = q, \\ \{a \in \Sigma \mid \delta(p, a) = q\}, & \text{sonst} \end{cases}$$



nur Buchstaben (und eventuell das leere Wort) und ist somit leicht durch einen regulären Ausdruck $\gamma_{p,q}^0$ beschreibbar. Wegen

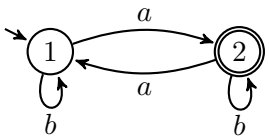
$$L_{p,q}^{r+1} = L_{p,q}^r \cup L_{p,r+1}^r (L_{r+1,r+1}^r)^* L_{r+1,q}^r$$

lassen sich aus den regulären Ausdrücken $\gamma_{p,q}^r$ für die Sprachen $L_{p,q}^r$ leicht reguläre Ausdrücke für die Sprachen $L_{p,q}^{r+1}$ gewinnen:

$$\gamma_{p,q}^{r+1} = \gamma_{p,q}^r | \gamma_{p,r+1}^r (\gamma_{r+1,r+1}^r)^* \gamma_{r+1,q}^r.$$

■

**Beispiel 25.** *Betrachte den DFA*




*Da $M$ insgesamt $m = 2$ Zustände und nur den Endzustand 2 besitzt, ist*

$$L(M) = \bigcup_{q \in E} L_{1,q} = L_{1,2} = L_{1,2}^2 = L(\gamma_{1,2}^2).$$

*Um $\gamma_{1,2}^2$ zu berechnen, benutzen wir die Rekursionsformel*

$$\gamma_{p,q}^{r+1} = \gamma_{p,q}^r | \gamma_{p,r+1}^r (\gamma_{r+1,r+1}^r)^* \gamma_{r+1,q}^r$$

*und erhalten*

$$\begin{aligned} \gamma_{1,2}^2 &= \gamma_{1,2}^1 | \gamma_{1,2}^1 (\gamma_{2,2}^1)^* \gamma_{2,2}^1, \\ \gamma_{1,2}^1 &= \gamma_{1,2}^0 | \gamma_{1,1}^0 (\gamma_{1,1}^0)^* \gamma_{1,2}^0, \\ \gamma_{2,2}^1 &= \gamma_{2,2}^0 | \gamma_{2,1}^0 (\gamma_{1,1}^0)^* \gamma_{1,2}^0. \end{aligned}$$

*Um den regulären Ausdruck $\gamma^2_{1,2}$ für $L(M)$ zu erhalten, genügt es also, die regulären Ausdrücke $\gamma^0_{1,1}$, $\gamma^0_{1,2}$, $\gamma^0_{2,1}$, $\gamma^0_{2,2}$, $\gamma^1_{1,2}$ und $\gamma^1_{2,2}$ zu berechnen:*

| $r$ | $p,q$ | | | |
|---|---|---|---|---|
| | $1,1$ | $1,2$ | $2,1$ | $2,2$ |
| 0 | $\epsilon \mid b$ | $a$ | $a$ | $\epsilon \mid b$ |
| 1 | - | $\underbrace{a \mid (\epsilon \mid b)(\epsilon \mid b)^* a}_{b^* a}$ | - | $\underbrace{(\epsilon \mid b) \mid a(\epsilon \mid b)^* a}_{\epsilon \mid b \mid ab^* a}$ |
| 2 | - | $\underbrace{b^* a \mid b^* a(\epsilon \mid b \mid ab^* a)^*(\epsilon \mid b \mid ab^* a)}_{b^* a(b \mid ab^* a)^*}$ | - | - |

◁

**Korollar 26.** *Sei $L$ eine Sprache. Dann sind folgende Aussagen äquivalent:*

- *$L$ ist regulär (d.h. es gibt einen DFA $M$ mit $L = L(M)$),*
- *es gibt einen NFA $N$ mit $L = L(N)$,*
- *es gibt einen regulären Ausdruck $\gamma$ mit $L = L(\gamma)$,*
- *$L$ lässt sich mit den Operationen Vereinigung, Produkt und Sternhülle aus endlichen Sprachen gewinnen,*
- *$L$ lässt sich mit den Operationen $\cap$, $\cup$, Komplement, Produkt und Sternhülle aus endlichen Sprachen gewinnen.*

Wir werden bald noch eine weitere Charakterisierung von REG kennenlernen, nämlich durch reguläre Grammatiken. Zuvor befassen wir uns jedoch mit dem Problem, DFAs zu minimieren. Dabei spielen Relationen (insbesondere Äquivalenzrelationen) eine wichtige Rolle.



## 2.4 Relationalstrukturen

Sei $A$ eine nichtleere Menge, $R_i$ eine $k_i$-stellige Relation auf $A$, d.h. $R_i \subseteq A^{k_i}$ für $i = 1, \ldots, n$. Dann heißt $(A; R_1, \ldots, R_n)$ ***Relationalstruktur***. Die Menge $A$ heißt ***Grundmenge***, ***Trägermenge*** oder ***Individuenbereich*** der Relationalstruktur.

Wir werden hier hauptsächlich den Fall $n = 1$, $k_1 = 2$, also $(A, R)$ mit $R \subseteq A \times A$ betrachten. Man nennt dann $R$ eine ***(binäre) Relation*** auf $A$. Oft wird für $(a, b) \in R$ auch die ***Infix-Schreibweise*** $aRb$ benutzt.

**Beispiel 27.**

- $(F, M)$ *mit* $F = \{f \mid f$ *ist Fluss in Europa*$\}$ *und*
$$M = \{(f, g) \in F \times F \mid f \text{ mündet in } g\}.$$
- $(U, B)$ *mit* $U = \{x \mid x$ *ist Berliner*$\}$ *und*
$$B = \{(x, y) \in U \times U \mid x \text{ ist Bruder von } y\}.$$
- $(\mathsf{P}(M), \subseteq)$, *wobei* $\mathsf{P}(M)$ *die Potenzmenge einer beliebigen Menge $M$ und $\subseteq$ die Inklusionsbeziehung auf den Teilmengen von $M$ ist.*
- $(A, Id_A)$, *wobei* $Id_A = \{(x, x) \mid x \in A\}$ *die* ***Identität auf $A$*** *ist.*
- $(\mathbb{R}, \leq)$.
- $(\mathbb{Z}, |)$, *wobei $|$ die "teilt"-Relation bezeichnet (d.h. $a|b$, falls ein $c \in \mathbb{Z}$ mit $b = ac$ existiert).* ◁

Da Relationen Mengen sind, sind auf ihnen die mengentheoretischen Operationen ***Schnitt***, ***Vereinigung***, ***Komplement*** und ***Differenz*** definiert. Seien $R$ und $S$ Relationen auf $A$, dann ist

$$\begin{aligned} R \cap S &= \{(x, y) \in A \times A \mid xRy \wedge xSy\}, \\ R \cup S &= \{(x, y) \in A \times A \mid xRy \vee xSy\}, \\ R - S &= \{(x, y) \in A \times A \mid xRy \wedge \neg xSy\}, \\ \overline{R} &= (A \times A) - R. \end{aligned}$$

Sei allgemeiner $\mathcal{M} \subseteq \mathcal{P}(A \times A)$ eine beliebige Menge von Relationen auf $A$. Dann sind der ***Schnitt über $\mathcal{M}$*** und die ***Vereinigung über $\mathcal{M}$*** folgende Relationen:

$$\bigcap \mathcal{M} = \bigcap_{R \in \mathcal{M}} R = \{(x,y) \mid \forall R \in \mathcal{M} : xRy\},$$
$$\bigcup \mathcal{M} = \bigcup_{R \in \mathcal{M}} R = \{(x,y) \mid \exists R \in \mathcal{M} : xRy\}.$$

Die ***transponierte (konverse) Relation*** zu $R$ ist

$$R^T = \{(y,x) \mid xRy\}.$$

$R^T$ wird oft auch mit $R^{-1}$ bezeichnet. Z.B. ist $(\mathbb{R}, \leq^T) = (\mathbb{R}, \geq)$.

Seien $R$ und $S$ Relationen auf $A$. Das ***Produkt*** oder die ***Komposition*** von $R$ und $S$ ist

$$R \circ S = \{(x,z) \in A \times A \mid \exists y \in A : xRy \wedge ySz\}.$$

**Beispiel 28.** *Ist $B$ die Relation "ist Bruder von", $V$ "ist Vater von", $M$ "ist Mutter von" und $E = V \cup M$ "ist Elternteil von", so ist $B \circ E$ die Onkel-Relation.* ◁

Übliche Bezeichnungen für das Relationenprodukt sind auch $R;S$ und $R \cdot S$ oder einfach $RS$. Das $n$-fache Relationenprodukt $R \circ \cdots \circ R$ von $R$ wird mit $R^n$ bezeichnet. Dabei ist $R^0 = Id$.

**Vorsicht:** Das $n$-fache Relationenprodukt $R^n$ von $R$ sollte nicht mit dem $n$-fachen kartesischen Produkt $R \times \cdots \times R$ der *Menge* $R$ verwechselt werden. Wir vereinbaren, dass $R^n$ das $n$-fache Relationenprodukt bezeichnen soll, falls $R$ eine Relation ist.

### Eigenschaften von Relationen

Sei $R$ eine Relation auf $A$. Dann heißt $R$

| | | |
|---|---|---|
| ***reflexiv***, | falls $\forall x \in A : xRx$ | (also $Id_A \subseteq R$) |
| ***irreflexiv***, | falls $\forall x \in A : \neg xRx$ | (also $Id_A \subseteq \overline{R}$) |
| ***symmetrisch***, | falls $\forall x,y \in A : xRy \Rightarrow yRx$ | (also $R \subseteq R^T$) |
| ***asymmetrisch***, | falls $\forall x,y \in A : xRy \Rightarrow \neg yRx$ | (also $R \subseteq \overline{R^T}$) |
| ***antisymmetrisch***, | falls $\forall x,y \in A : xRy \wedge yRx \Rightarrow x = y$ | (also $R \cap R^T \subseteq Id$) |
| ***konnex***, | falls $\forall x,y \in A : xRy \vee yRx$ | (also $A \times A \subseteq R \cup R^T$) |
| ***semikonnex***, | falls $\forall x,y \in A : x \neq y \Rightarrow xRy \vee yRx$ | (also $\overline{Id} \subseteq R \cup R^T$) |
| ***transitiv***, | falls $\forall x,y,z \in A : xRy \wedge yRz \Rightarrow xRz$ | (also $R^2 \subseteq R$) |

gilt.

Die nachfolgende Tabelle gibt einen Überblick über die wichtigsten Relationalstrukturen.

| | refl. | sym. | trans. | antisym. | asym. | konnex | semikon. |
|---|---|---|---|---|---|---|---|
| Äquivalenzrelation | ✓ | ✓ | ✓ | | | | |
| (Halb-)Ordnung | ✓ | | ✓ | ✓ | | | |
| Striktordnung | | | ✓ | | ✓ | | |
| lineare Ordnung | | | ✓ | ✓ | | ✓ | |
| lin. Striktord. | | | ✓ | | ✓ | | ✓ |
| Quasiordnung | ✓ | | ✓ | | | | |

In der Tabelle sind nur die definierenden Eigenschaften durch ein "✓" gekennzeichnet. Das schließt nicht aus, dass gleichzeitig auch noch weitere Eigenschaften vorliegen können.

**Beispiel 29.**

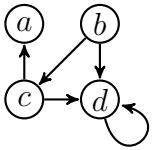
- *Die Relation "ist Schwester von" ist zwar in einer reinen Damengesellschaft symmetrisch, i.a. jedoch weder symmetrisch noch asymmetrisch noch antisymmetrisch.*

- *Die Relation "ist Geschwister von" ist zwar symmetrisch, aber weder reflexiv noch transitiv und somit keine Äquivalenzrelation.*
- $(\mathbb{R}, <)$ *ist irreflexiv, asymmetrisch, transitiv und semikonnex und somit eine lineare Striktordnung.*
- $(\mathbb{R}, \leq)$ *und* $(\mathsf{P}(M), \subseteq)$ *sind reflexiv, antisymmetrisch und transitiv und somit Ordnungen.*
- $(\mathbb{R}, \leq)$ *ist auch konnex und somit eine lineare Ordnung.*
- $(\mathsf{P}(M), \subseteq)$ *ist zwar im Fall* $\|M\| \leq 1$ *konnex, aber im Fall* $\|M\| \geq 2$ *weder semikonnex noch konnex.* ◁

### Graphische Darstellung von Relationen

Eine Relation $R$ auf einer endlichen Menge $A$ kann durch einen ***gerichteten Graphen*** (oder ***Digraphen***) $G = (V, E)$ mit ***Knotenmenge*** $V = A$ und ***Kantenmenge*** $E = R$ veranschaulicht werden. Hierzu stellen wir jedes Element $x \in A$ als einen Knoten dar und verbinden jedes Knotenpaar $(x, y) \in R$ durch eine gerichtete Kante (Pfeil). Zwei durch eine Kante verbundene Knoten heißen ***benachbart*** oder ***adjazent***.

**Beispiel 30.** *Für die Relation* $(A, R)$ *mit* $A = \{a, b, c, d\}$ *und* $R = \{(b, c), (b, d), (c, a), (c, d), (d, d)\}$ *erhalten wir folgende graphische Darstellung.*




◁

Der ***Ausgangsgrad*** eines Knotens $x \in V$ ist $\deg^+(x) = \|R[x]\|$, wobei $R[x] = \{y \in V \mid xRy\}$ die Menge der ***Nachfolger*** von $x$ ist. Entsprechend ist $\deg^-(x) = \|\{y \in V \mid yRx\}\|$ der ***Eingangsgrad*** von $x$ und $R^{-1}[x] = \{y \in V \mid yRx\}$ die Menge der ***Vorgänger*** von $x$. Falls $R$ symmetrisch ist, werden die Pfeilspitzen meist weggelassen. In diesem Fall ist $d(x) = \deg^-(x) = \deg^+(x)$ der ***Grad*** von $x$ und $R[x] = R^{-1}[x]$ heißt die ***Nachbarschaft*** von $x$. Ist $R$ zudem irreflexiv, so ist $G$ ***schleifenfrei*** und wir erhalten einen ***(ungerichteten) Graphen***. Eine irreflexive und symmetrische Relation $R$ wird meist als Menge der ungeordneten Paare $E = \big\{\{a, b\} \mid aRb\big\}$ notiert.