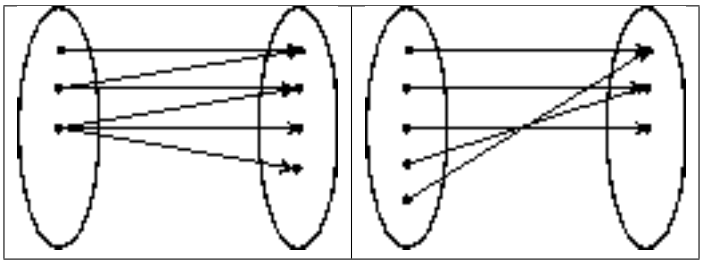
1.5. Abbildung

Eine *Abbildung* (oder *Funktion*) ist eine Relation f über $X \times Y$ mit der Eigenschaft: für jedes x aus X gibt es genau ein $y \in Y$ mit $(x, y) \in f$. Die übliche Schreibweise ist $f : X \to Y$. Dabei wird das eindeutige y welches zu x existiert mit $f(x)$ bezeichnet. Als Relationssymbol wird $\mapsto$ verwendet. Verlangt man anstelle von 'genau ein' 'höchstens ein' spricht man von partiellen Funktionen. X heißt *Definitionsbereich* und Y *Bildbereich* der Abbildung f. Wird ein $x \in X$ unter f auf y abgebildet, so ist y das *Bild* von x (unter f) und x ist ein *Urbild* von y.

EXAMPLE 1.5.1. Vergleich Relation und Abbildung

Der Unterschied ist, dass bei einer Relation einem Punkt aus X mehrere Punkte aus Y zugeordnet werden können:



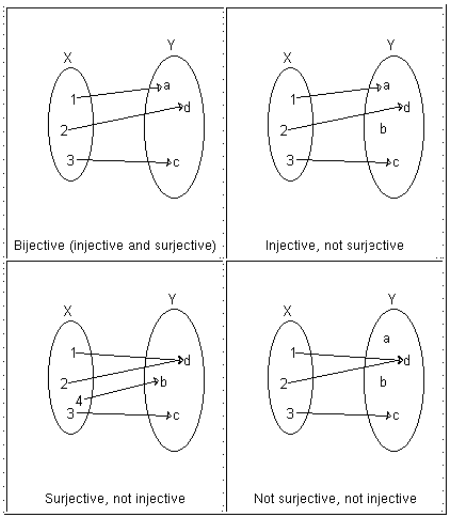
DEFINITION 1.5.2. injektiv, surjektiv, bijektiv

Eine Abbildung f ist *injektiv*, falls es zu jedem $y \in Y$ höchstens ein $x \in X$ gibt mit $y = f(x)$.

Eine Abbildung f ist *surjektiv*, falls es zu jedem $y \in Y$ mindestens ein $x \in X$ mit $y = f(x)$.

Eine Abbildung f ist *bijektiv*, falls sie injektiv und surjektiv ist.

EXAMPLE 1.5.3. Bilder aus Wikipedia



Die inverse Relation f^{-1} einer Abbildung ist im Allgemeinen keine Abbildung, nur im Falle einer Bijektion ist f^{-1} eine Abbildung. Man nennt f^{-1} auch *Umkehrabbildung*. Bei einer bijektiven Abbildung ergibt die Komposition $f^{-1} \circ f$ die *identische* Abbildung $x \mapsto x$. Bijektive Funktionen werden in der Mengenlehre (wie gesehen) verwendet um die Kardinaltät von Mengen zu vergleichen. Zwei Mengen X, Y haben gleiche Kardinalität falls zwischen ihnen ein Bijektion gibt. Wir haben dies gesehen beim Begriff der Abzählbarkeit einer Menge X, dies bedeutet es gibt eine Bijektion zwischen X und $\mathbb{N}$.

1.5.1. Abbildungen zwischen endlichen Mengen. Abbildungen $f : X \to Y$ zwischen endlichen Mengen $X = \{x_1, \ldots, x_m\}$ und $Y = \{y_1, \ldots, y_m\}$ kann man mit $m-$Tupeln identifizieren, der Eintrag an der $i-$ten Komponente ist das Element aus Y auf das x_i abgebildet wird. Damit wird klar

LEMMA 1.5.4. *Anzahl der Abbildungen zwischen endlichen Mengen*

Seien X und Y endliche Mengen, dann ist die Anzahl der Abbildungen $f : X \to Y$:

$$|Y|^{|X|}.$$

BEWEIS. Man überlegt sich wieviel verschiedene Tupel es gibt. □

Das erklärt auch, warum man die Notation Y^X für die Menge aller Abbildungen von X nach Y verwendet. Dies ist auch kompatibel mit der Notation 2^X für die Potenzmengen, denn der charakteristische Vektor ist als Tupel eine Abbildung von X nach $\{0, 1\}$.

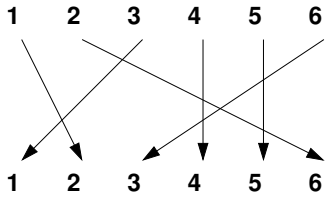
LEMMA 1.5.5. *Abbildungen zwischen gleichmächtigen endlichen Mengen*

Seien X, Y gleichmächtige endliche Mengen, d.h. $|X| = |Y|$. Dann gilt:

- *Jede surjektive Abbildung $X \to Y$ ist bereits bijektiv.*
- *Jede injektive Abbildung $X \to Y$ ist bereits bijektiv.*

1.5.2. Permutationen. Bijektive Abbildungen $\pi : X \to X$ werden als *Permutationen* bezeichnet (man nimmt meist griechische Buchstaben). Meist beschränkt man sich dabei auf endliche Mengen $X = \{1, 2, \ldots, n\}$. Die Mächtigkeit n ist der *Grad* der Permutation. Die Tupelschreibweise heißt bei Permutationen Listenschreibweise. Es gibt noch eine erweitere Listenschreibweise, dabei wird in einer extra Zeile das jeweilige Urbild (also $1, 2, \ldots, n$) notiert.

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 6 & 1 & 4 & 5 & 3 \end{pmatrix} = \quad [2, 6, 1, 4, 5, 3] =$$



Die inverse Permutation (da Bijektion, ist dies wieder eine Permutation) erhält man durch Vertauschen der beiden Zeilen in der erweiterten Listenschreibweise und Sortieren der obersten Zeile:

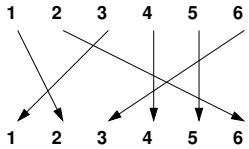
$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 6 & 1 & 4 & 5 & 3 \end{pmatrix} \to \begin{pmatrix} 2 & 6 & 1 & 4 & 5 & 3 \\ 1 & 2 & 3 & 4 & 5 & 6 \end{pmatrix} \to \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 1 & 6 & 4 & 5 & 2 \end{pmatrix} = \pi^{-1}$$

Die Komposition zweier Permutation lässt sich einfach bestimmen, wenn man die erweiterte Listenschreibweise verwendet:

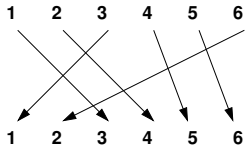
$$\tau \circ \pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 1 & 5 & 6 & 2 \end{pmatrix} \circ \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 6 & 1 & 4 & 5 & 3 \end{pmatrix} = ?$$

Um diese Komposition zu berechnen kann man die Spalten der zweiten Permutation so vertauschen, dass die oberste Zeile von τ der Reihenfolge der untersten Zeile von π entspricht, die unterste Zeile von τ ist dann die Komposition. Man kann das Ergebnis auch direkt ablesen, oder sich auch mit den Weg durch die zusammengesetzten Bilder helfen:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 1 & 5 & 6 & 2 \end{pmatrix} \circ \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 6 & 1 & 4 & 5 & 3 \end{pmatrix} =?$$



$$= \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 6 & 1 & 4 & 5 & 3 \end{pmatrix}$$



$$= \begin{pmatrix} 2 & 6 & 1 & 4 & 5 & 3 \\ 4 & 2 & 3 & 5 & 6 & 1 \end{pmatrix} \qquad = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 1 & 5 & 6 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 4 & 2 & 3 & 5 & 6 & 1 \end{pmatrix}$$

Man sieht die Komposition gerne als Multiplikation, und schreibt dann $\tau\pi$ anstelle von $\tau \circ \pi$. Man muss nur in jedem neuen Mathematikbuch wieder extra nachschauen, denn es gibt auch Autoren die dann nicht von rechts nach links multiplizieren sondern von links nach rechts. Das ist leider ein Problem, denn die Komposition ist nicht kommutativ:

EXAMPLE 1.5.6. Multiplikation von Permutation ist nicht kommutativ

Obiges Beispiel zeigt dies bereits:

$$\tau\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 1 & 5 & 6 & 2 \end{pmatrix} \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 6 & 1 & 4 & 5 & 3 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 4 & 2 & 3 & 5 & 6 & 1 \end{pmatrix}$$

$$\pi\tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 6 & 1 & 4 & 5 & 3 \end{pmatrix} \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 1 & 5 & 6 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 4 & 2 & 5 & 3 & 6 \end{pmatrix}$$

und damit sieht man

$$\tau\pi \neq \pi\tau$$

LEMMA 1.5.7. *Anzahl der Permutationen*

Die Anzahl der Permutationen vom Grad n (=Anzahl der bijektiven Abbildungen zwischen zwei Mengen der Kardinalität n) ist

$$n!.$$

BEWEIS. Man zählt mögliche Tupel. □

1.6. Automaten

Alan Turing (23. 6. 1912- 7. 6. 1954)

Im Abschnitt 1.1 über die Mengenlehre haben wir gesehen, dass es vielmehr Funktionen $\mathbb{N} \to \mathbb{N}$ gibt als wir mit einem Programm ausrechnen können. Dies war aus reinen Anzahlgründen so. Nun kann man sich natürlich fragen welche Funktionen kann ich berechnen und welche nicht. Zur Präzessierung dieser Fragestellung definierte 1937 Alan Turing die später sogenannte Turing Maschine, die ein mathematisches Modell eines damals noch nicht existierenden Computers darstellt. Ein eingeschränktes Modell wird im Folgenden betrachtet. Damit wird nicht eine beliebige Funktion implementiert sondern nur ein Entscheider, der ja/nein bzw. 0/1 als Ergebnis liefert.

EXAMPLE 1.6.1. parity check

Als erstes Beispiel soll getestet werden ob eine 0/1 Folge eine gerade Anzahl von Einsen hat. Dies ist zum Beispiel bei Speicherbausteinen von Interesse (Parity-Check-Memory). Unsere Maschine hat also als Eingabe eine 0/1 Folge und als Ausgabe ja/nein. Dies kann wie folgt realisiert werden. Die Maschine hat zwei Zustände (G(=gerade Anzahl bisher gelesen) und U). Zu Beginn ist die Maschine im Zustand G, da eine Folge der Länge 0 eine gerade Anzahl von Einsen hat. Nun wird die 0/1 Folge eingelesen und je nach eintreffenden Zeichen wird der Zustand gewechselt (bei eingelesen Eins) oder nicht. Ist nach dem Lesen des Wortes der Zustand G, dann hat die 0/1 Folge eine gerade Anzahl von Einsen. Dies visualisiert man am besten mit dem Übergangsgraphen:

$$\mathrm{G} \overset{1}{\to} \mathrm{U} \overset{0}{\to} \mathrm{U} \overset{1}{\to} \mathrm{G} \overset{0}{\to} \mathrm{G}$$

Da nach Lesen der Eingabefolge 1010 der Zustand G erreicht ist wissen wir, dass diese 4-stellige Eingabefolge eine gerade Anzahl von Einsen hat. Im Kontext von Parity-Check-Memory würde die 4-Bit Speicherbelegung als korrekt erkannt, wohingegen eine Speicherbelegung 1011 als fehlerhaft erkannt würde.

DEFINITION 1.6.2. endlicher Automat

Ein endlicher *Automat* ist ein 5-Tupel $(S, \Sigma, S_0, \delta, F)$ mit einer endlichen Zustandsmenge S, einem endlichen Alphabet (=Eingabezeichen) Σ, einem Startzustand $S_0 \in S$, einer Übergangsfunktion $\delta : S \times \Sigma \to S$, und einer Endzustandsmenge $F \subset S$.

Ein endlicher Automat *akzeptiert* ein Wort (=Eingabefolge mit Zeichen aus Σ) falls nach Einlesen des Wortes der Automat in einem akzeptieren Zustand (d.h. in einem Zustand aus F) ist.

EXAMPLE 1.6.3. parity check

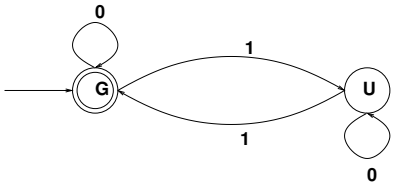
$S = \{G, U\}; \Sigma = \{0, 1\}; F = \{G\}; S_0 = G;$

Die Übergangsfunktion δ in tabellarischer Form:

$s \in S$	$\sigma \in \Sigma$	$\delta(s, \sigma)$
G	0	G
G	1	U
U	0	U
U	1	G

Die gesamte Information wird im Zustandsgraph visualisiert, dabei sind die Zustände Knoten, und gerichtete mit einem Eingabezeichen beschriftete Kanten geben den Zustandswechsel wieder. Der Startzustand wird mit einen Pfeil ohne Startknoten markiert. Akzeptierende Zustände werden durch Doppelkreise markiert.

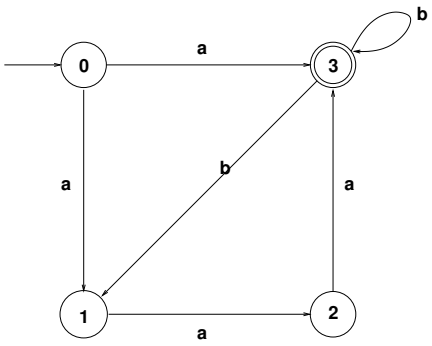
EXAMPLE 1.6.4. parity check Zustandsgraph



Das Konzept der endlichen Automaten findet ihre Hauptanwendung in der Theorie der formalen Sprachen. Man bezeichnet als *Sprache* des Automaten die Menge der Worte (über dem Eingabealphabet) die zu einem akzeptierenden Zustand führen. Umgekehrt versucht man zu einer Sprache einen endlichen Automaten zu basteln. Dabei erweitert man die Definition des Automaten zu einem *nicht deterministischen endlichen Automaten* (NEA) indem man die Übergangsfunktion δ durch eine Übergangsrelation ersetzt, d.h. nun kann es zu einem Paar (s, σ) mehrere oder auch keine Nachfolgezustände haben. Eine typische Fragestellung ist dann die Konstruktion eines NEA zum Erkennen eines regulären Ausdrucks.

EXAMPLE 1.6.5. NEA

Gesucht wird ein Automat für die reguläre Sprache $(a|aaa)(b^*baa)^*b^*$ das sind alle Worte die entweder mit a oder aaa beginnen, danach können mehrere Teile mit mindestens einem b und zwei a kommen, und zum Schluss können noch beliebig viele b kommen. Ein zugehöriger NEA kann wie folgt aussehen:



Die zugehörige Übergangsrelation ist (als $0/1$ Matrix)

	0	1	2	3
$(0, a)$		1		1
$(0, b)$				
$(1, a)$			1	
$(1, b)$				
$(2, a)$				1
$(2, b)$				
$(3, a)$				
$(3, b)$		1		1

man sieht, dass es sowohl Zeilen gibt mit mehreren Nachfolgern, als auch Paare $(s, \sigma) \in S \times \Sigma$ die keinen Nachfolger haben. Will man dieses Verfahren wirklich zum Erkennen implementieren wendet man dann einen Algorithmus an um den NEA in einen (deterministischen) endlichen Automaten umzuwandeln. (siehe spätere Vorlesung formale Sprachen und Compilerbau)