# Graphen, Kategorien, Funktoren (5)

**Definition:** Ein kreisfreier (orientierter) Graph mit Stern heißt Baum. Sterne eines Baumes heißen Wurzeln des Baumes.

**Lemma:** Zu jedem Graphen $G = (E, K)$ mit Stern $S$ gibt es einen Baum mit $K$ als Knotenmenge, der Untergraph von $G$ ist und $S$ als Wurzel hat.

**Definition:** Ein Baum, wie er in dem vorhergehenden Lemma beschrieben wird, heißt **aufspannender** Baum von $G$.

**Lemma:** Die aufspannenden Bäume nicht orientierter Graphen lassen sich durch eine Folge von Kantenaustauschschritten ineinander transformieren.

# Graphen, Kategorien, Funktoren (6)

**Beweis:** Übung.

**Hinweise:**

(1) Definiere Distanz von aufspannenden Bäumen

(2) Zeige, daß sich die Distanz verringern läßt.

**Beobachtung:** Die Wurzel und die fehlenden Kanten (von $G$) bestimmen den jeweiligen Baum eindeutig.

# Wegekategorie (1)

**Die Kategorie der Wege von $G$**

Sind $w_1, w_2 \in \mathcal{W}(G)$ und ist $Q(w_2) = Z(w_1)$

$w_1 = (Q_1, s_1, \ldots, s_n, Z_1)$, $w_2 = (Q_2, t_1, \ldots, t_m, Z_2)$,

dann ist wegen $Z_1 = Z(w_1) = Q(w_2) = Q_2$ auch

$$w := (Q_1, s_1, \ldots, s_n, t_1, \ldots, t_m, Z_2)$$

ein Weg aus $\mathcal{W}(G)$.

**Definition:** Das Produkt $w_1 \cdot w_2$ zweier Wege $w_1, w_2$ ist genau dann definiert, wenn $Z(w_1) = Q(w_2)$ ist und es gilt $w_1 \cdot w_2 := w$.

**Lemma:** Das Produkt von Wegen ist, soweit definiert, assoziativ. Für $1_{Q(w)}$ und $1_{Z(w)}$ gilt

$$1_{Q(w)} \cdot w = w = w \cdot 1_{Z(w)}.$$

**Beweis:** Klar.

## **Wegekategorie** (2)

$(\mathcal{W}(G), \cdot, Q, Z)$ ist eine **Kategorie**.

**Definition:**

(1) $w \cdot v$, $w \cdot u$ und $r \cdot v$ definiert, dann auch $r \cdot u$ definiert.

(2) Zu jedem $w \in \mathcal{W}$ gibt es eine Linkseins und eine Rechtseins.

## **Wegekategorie** (3)

**Definition:** Eine Menge $\mathcal{K}$ mit einer partiellen Verknüpfung $\cdot$ heißt Kategorie $:\Longleftrightarrow$ Es gelten

(1) Sind die Produkte $(w_1 \cdot w_2) \cdot w_3$ und $w_1 \cdot (w_2 \cdot w_3)$definiert, dann gilt

$$(w_1 \cdot w_2) \cdot w_3 = w_1 \cdot (w_2 \cdot w_3).$$

(2) Zu jedem $w$ gibt es eine Linkseins $1_{l(w)}$ und $1_{r(w)}$ mit

$$1_{l(w)} \cdot w = w = w \cdot 1_{r(w)}.$$

(3) Ist $w \cdot v$ definiert, dann ist $1_{r(w)} = 1_{l(v)}$.

**Bemerkung:** Anstelle von $l(w)$ schreibt man üblicherweise $Q(w)$ und anstelle von $r(w)$ $Z(w)$.

## **Wegekategorie** (4)

Der Begriff des Homomorphismus überträgt sich in natürlicher Weise.

(1) $\varphi(w \cdot v) = \varphi(w) \cdot \varphi(w)$

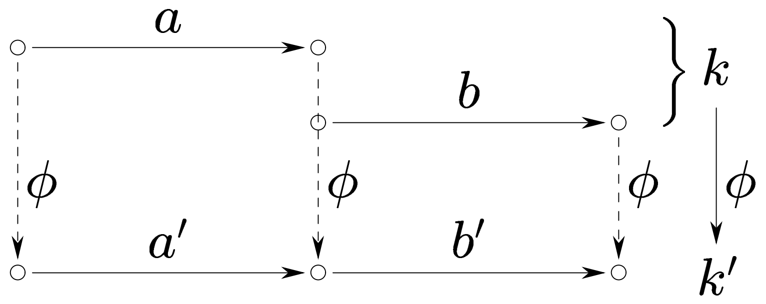
(2) $\varphi(1_{l(w)}) = 1_{l(\varphi(w))}$, $\varphi(1_{r(w)}) = 1_{r\varphi(w)}$.

Im Zusammenhang mit Kategorie spricht man aber nicht von Homomorphismen, sondern von Funktoren.

**Grund:** Ein für Homomorphismen fundamentaler Satz gilt nicht. $\varphi(\mathcal{K})$ ist nicht notwendig eine Kategorie.

# **Wegekategorie** (5)

**Beispiel:**




$a \cdot b$ ist **nicht** definiert in $\mathcal{K}$.

Für $a' = \varphi(a)$ und $b' = \varphi(b)$ ist $a' \cdot b'$ definiert.

$\varphi(\mathcal{K})$ enthält $a' \cdot b'$ nicht.

## **Wegekategorie** (6)

Die Definition der „Freiheit“ von Monoiden überträgt sich auf Kategorien. Dem Wortmonoid entspricht (als Verallgemeinerung) die Wegekategorie. Es gilt für jede Kategorie $\mathcal{K}$ :

$$\varphi' : E \cup \{1_P \mid P \in K\} \to \mathcal{K}$$

Jede Abbildung $\varphi'$, die $\varphi'(1_{l(w)}) = 1_{l\varphi'(w)}$ und $\varphi'(1_{r(w)}) = 1_{r(\varphi'(w))}$ erfüllt, läßt sich auf genau eine Weise zu einem Funktor $\varphi : \mathcal{W}(G) \to \mathcal{K}$ fortsetzen.

Die Voraussetzung ist immer erfüllt, wenn $\mathcal{K}$ ein Monoid $M$ ist.

**Bezeichnung:** $\varphi : \mathcal{W}(G) \to M$ heißt Beschriftung von $\mathcal{W}(G)$ mit Elementen aus $M$.

# Endliche Akzeptoren (1)

**Definition:** $a = (\mathcal{W}(G), S, T, \mu, X)$ heißt (nicht deterministischer) Akzeptor, falls $S, T \subset K(G)$ und $\mu : \mathcal{W}(G) \to X^*$ ein Funktor in das freie Monoid $X^*$ ist. $S$ heißt die Menge der Startknoten und $T$ die Menge der Terminalknoten. $\mathcal{W}(G)(S,T) = \{w \in \mathcal{W}(G) \mid Q(w) \in S \text{ und } Z(w) \in T\}$, ist die Menge der Wege von $S$ nach $T$ in $G$.

Die Menge

$$L_a = \{\mu(w) \mid w \in \mathcal{W}(G)(S,T)\}.$$

heißt die durch $a$ akzeptierte Sprache.

**Beispiel: Akzeptor für die Dycksprache**

$K := synt(L)$,

$E := \{([u], \sigma_i, [u\sigma_i]) \mid u \in \Sigma^*, i = 1, \ldots, n\}_n \cup \{([u], \sigma_i^-, [u\sigma_i^-]) \mid u \in \Sigma^*, i = 1, \ldots, u\}.$

# Endliche Akzeptoren (2)

Die Menge der Kanten ist also repräsentiert durch

$$[u] \xrightarrow{\sigma_i} [u\sigma_i],\ \ [u] \xrightarrow{\sigma_i^-} [u\sigma_i^-].$$

$G = (E, K, S)$;

$$\varrho([u], \sigma_i, [u\sigma_i]) = ([u], [u\sigma_i]).$$

Der Graph hat eine sehr einfache Struktur:

$$\begin{tikzcd}
[1] \arrow[r, "\sigma_{i_1}"] \arrow[dd, "\sigma_{j_1}^-"] & {[\sigma_{i_1}]} \arrow[r, "\sigma_{i_2}"] \arrow[l, bend left, "\sigma_{i_1}^-"] \arrow[dr, "\sigma_i^- \neq \sigma_{i_1}^-"] & \cdots \arrow[r, "\sigma_{i_n}"] & {[\sigma_{i_1} \ldots \sigma_{i_n}]} \arrow[r, "\sigma_i"] \arrow[dl, "\sigma_l^- \neq \sigma_{i_n}^-"] & {[\sigma_{i_1} \ldots \sigma_{i_n}\sigma_i]} \arrow[l, bend left, "\sigma_i^-"] \\
 & & {[0]} & & \\
{[\sigma_{j_1}^-]} \arrow[r] & \cdots \arrow[r] & {[\sigma_{j_1}^- \ldots \sigma_{j_m}^-]} \arrow[r, "\sigma^-"] \arrow[d, "\sigma_i"] & {[\sigma_{j_1}^- \ldots \sigma_{j_m}^- \sigma^-]} & \\
 & & {[\sigma_{j_1}^- \ldots \sigma_{j_m}^- \sigma_i]} \arrow[r] & \cdots &
\end{tikzcd}$$

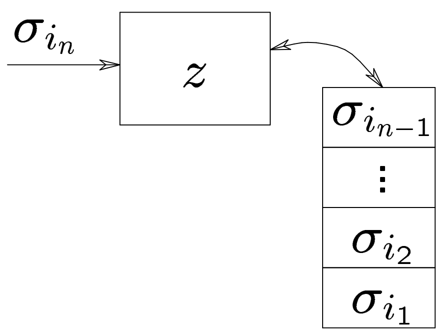
# Endliche Akzeptoren (3)

Man sieht, daß der Zweig des Graphen, der mit einem $\sigma_i^-$ beginnt, keinen (orientierten) Weg zu $T = [1]$ enthält. Damit ist dieser Zweig für den Akzeptor überflüssig.

Erläuterung durch Klammern

$$\begin{tikzcd}
{} \arrow[r] & ([((([ \arrow[r, "]"] \arrow[d, ")"] & ([((( \arrow[r] \arrow[d, "]"] & {} \\
& 0 & 0 &
\end{tikzcd}$$

$synt(D(\Sigma))$ repräsentiert die einfachste Form eines Kellerautomaten.




$z$ Zustand 1 weiter offen.

$z = 0$ Ablehnung.

## **Endliche Akzeptoren** (4)

Endlicher Automat

$a = (X, Z, Y, \delta, \lambda, z_0),\ z_0 \in Z$ (Anfangszustand)

$$\delta \subset X \times Z \times Z,\ \lambda \subset X \times Z \times Y$$

$a$ ist deterministisch, wenn $\delta : X \times Z \to Z$ und $\lambda : X \times Z \to Y$ Abbildungen sind.

Repräsentationen von $a$ durch einen Graphen
Knotenmenge $Z$. Kantenmenge $(x, z_1, z_2) \in \delta$

**Graphik s. Tafel!**

$y$ bezeichnet die Ausgabe:

$$(x, z, y) \in \lambda.$$

## **Endliche Akzeptoren** (5)

Indem man $Z' = Z \times Y$ setzt und

$$\begin{aligned}\tilde{\delta} = \ & \{((x, y_1), (z_2, y_2)) \mid (x, z_1, z_2) \in \delta \\ & \text{und } (x, z_1, y_2) \in \lambda\}\end{aligned}$$

$$\tilde{\lambda} \subset Z'$$

kann man formal die Ausgabe nur vom Zustand abhängig machen.

Nun definiert man

$$T_y = \{z \mid (z, y) \in \tilde{\lambda}\}.$$

Für jede Ausgabe wählt man eine Menge von Terminalzuständen. Das ist äquivalent zur Originaldefinition.

Somit kann man sich in der Theorie der endlichen Automaten weitgehend auf die Theorie der endlichen Akzeptoren zurückziehen. Das gilt zumindest soweit, als man sich nur für die grundsätzliche Leistungsfähigkeit der Automaten interessiert.

## **Endliche Akzeptoren** (6)

**Definition:** $REG(X^*) = \{L \subset X^* \mid \text{es gibt ein } a = (G, S, T, X, \mu)\}$, so daß

$$L = \mu(\mathcal{W}(G)(S, T)).$$

$REG(X^*)$ heißt die Menge der **regulären Mengen**.

Wir haben gesehen, daß

$$\begin{array}{ll} L = h^{-1}(T), & h : X^* \to synt(L) \\ & T = \{[a]_L \mid a \in L\}. \end{array}$$

Ist $L$ endlich, so ist auch $synt(L)$ endlich.

**Frage:** Was kann man über $L$ aussagen, wenn $synt(L)$ endlich ist?

**Frage:** Kann man die endlichen Akzeptoren aus einfacheren Akzeptoren aufbauen?

Wie sehen die regulären Sprachen aus? Kann man sie aus wenigen einfachen reg. Sprachen konstruieren?