

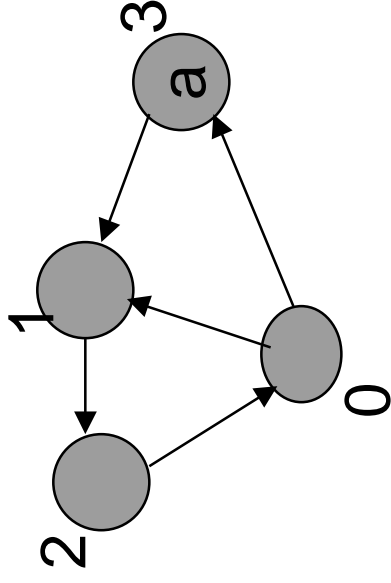
# Computergestützte Verifikation

9.5.03

# Stark zusammenhängende Mengen

## und starke Fairness

$$(0\ 1\ 2)^* \models (G\ F\ a) \Rightarrow (G\ F\ b)$$



kein Pfad, der unendlich oft 3 durchläuft, erfüllt die Annahme

→ es reicht nicht, *Komponenten* zu betrachten;  
alle *stark zusammenhängenden Mengen* müssen untersucht werden

immerhin: Jede SZM ist in einer SZK enthalten

## 3.5 Symmetrie

1. Formale Definition: symmetrisches Verhalten
2. Konstruktion des Quotientensystems
3. Welche Eigenschaften bleiben erhalten?
4. Erkennung von Datenstruktursymmetrie
5. Erkennung von Komponentensymmetrie

# 3.5.1 Symmetrisches Verhalten

Basis: Transitionssystem

$\sigma$  heißt Symmetrie, wenn:

-  $\sigma$  ist Bijektion  $S \rightarrow S$

-  $s - a \rightarrow s'$  gdw. ex.  $a'$ :  $\sigma(s) - e' \rightarrow \sigma(s')$

-  $s \in I$  gdw.  $\sigma(s) \in I$

$\Sigma_{TS}$ : Menge aller  
Symmetrien von TS

mit Induktion:

$s_0 s_1 s_2 \dots$  Pfad in einem Transitionssystem  $\rightarrow$

$\sigma(s_0) \sigma(s_1) \sigma(s_2) \dots$  ebenfalls

-Id ist immer Symmetrie

-Wenn  $\sigma$  Symmetrie, so auch  $\sigma^{-1}$

-Wenn  $\sigma_1$  und  $\sigma_2$  Symmetrien, so auch  $\sigma_1 \circ \sigma_2$

}  $[\Sigma_{TS}, \circ]$  ist  
Gruppe<sub>4</sub>

# Zustandsäquivalenz

Analyse der Struktur liefert i.d.R. nicht alle Symmetrien, aber immer eine bzgl. Inversion und Komposition abgeschlossene Menge von Symmetrien

→ ab jetzt betrachten wir immer eine beliebige Untergruppe  $\Sigma \subseteq \Sigma_{TS}$

$s \sim s'$  gdw. es ex.  $\sigma \in \Sigma$  mit  $\sigma(s) = s'$

$\sim$  ist Äquivalenzrelation

# Reduziertes Transitionssystem

$$TS_{\Sigma} = [S/\sim, E_{\Sigma}, I/\sim^*, ]$$

$$E_{\Sigma} = \{ [ [s],[s'] ] \mid \text{ex. } s \in [s], \text{ ex. } s' \in [s'] : [s,s'] \in E \}$$

Größe des reduzierten Systems:

$$|S/\sim| \geq |S| / |\Sigma|$$

$|\Sigma|$  kann exponentiell in der Zahl der Elemente eines symm. Datentyps bzw. exponentiell in der Anzahl der replizierten Komponenten sein

# 3.5.4 Symmetrie in Datentypen

Fall 1: Skalare Datentypen

- Menge  $D$  von Werten
- nur  $=$ ,  $\neq$  in Guards
- $:=$  (Zuweisung)
- als Indexmenge von (einfachen) Arrays anderer Datentypen
- Schleifen der Form FOR ALL  $x \in D$  DO ...
- choose( $x$ )
- keine Konstanten

Seien  $x_1, \dots, x_n$  alle Variablen eines skalaren Datentyps  $D$ ,  
 $\beta$  eine Belegung dieser Variablen mit Werten, und  $\pi$   
eine Permutation auf  $D$ .

Setzen  $\pi$  zu einer Symmetrie  $\sigma$  fort

# Symmetrie in Datentypen

$\sigma(s)$  :  $\sigma(s)(x) = \pi(s(x))$ , falls  $x$  vom Typ  $D$   
 $\sigma(s)(x[\pi(i)]) = s(x[i])$ , falls  $x$  array mit  $D$  als Indexmenge  
 $\sigma(s)(x) = s(x)$ , sonst

Beispiel:  $D = \{1,2,3,4\}$        $\pi: 1 \rightarrow 2$     $2 \rightarrow 4$     $3 \rightarrow 3$     $4 \rightarrow 1$

$s:$	$x1 = 1$	$\sigma(s):$	$x1 = 2$
	$x2 = 3$		$x2 = 3$
	$x3 = \text{"hallo"}$		$x3 = \text{"hallo"}$
	$x4[1] = A$		$x4[1] = C$
	$x4[2] = B$		$x4[2] = A$
	$x4[3] = C$		$x4[3] = C$
	$x4[4] = C$		$x4[4] = B$

# Nachweis der Symmetrieeigenschaft

Beispiel:  $D = \{1, 2, 3, 4\}$

$\pi: 1 \rightarrow 2 \quad 2 \rightarrow 4 \quad 3 \rightarrow 3 \quad 4 \rightarrow 1$

s:  $x_1 = 1$                        $\sigma(s): x_1 = 2$

$x_2 = 3$

$x_3 = \text{"hallo"}$

$x_4[1] = A$

$x_4[2] = B$

$x_4[3] = C$

$x_4[4] = C$

$x_1 = 2$

$x_2 = 3$

$x_3 = \text{"hallo"}$

$x_4[1] = C$

$x_4[2] = A$

$x_4[3] = C$

$x_4[4] = B$

1. Alle Terme anderer Datentypen liefern in  $s$  und  $\sigma(s)$  die gleichen Werte  
 $x_3 \quad x_4[x_1]$
2. Wenn ein Term in  $s$  den Wert  $d \in D$  liefert, so liefert er in  $\sigma(s)$  den Wert  $\pi(d)$   
 $x_1 \quad x_2$
3. Ausdrücke sind wahr in  $s$  gdw. in  $\sigma(s)$      $x_1 = x_2$      $x_4[x_1] \neq A$
4. Zuweisungen erhalten Eigenschaften 1-3

Also: Wenn  $s \rightarrow s'$ , so  $\sigma(s) \rightarrow \sigma(s')$

# Das Orbit-Problem

geg.: Zustand  $s$ , Zustandsmenge  $S$ , Symmetriegruppe  $\Sigma$

Frage: gibt es ein  $\sigma \in \Sigma$  mit  $\sigma(s) \in S$  ?

Lösung (meistens): kanonische Repräsentanten

= ein speziell ausgezeichneteter, leicht berechenbarer Vertreter der eigenen Äquivalenzklasse

- in  $S$  werden nur kanonische Repräsentanten gespeichert
- Orbitproblem reduziert sich zu:  $\text{canrep}(s) \in S$ ?

# Implementation von canrep auf skalaren Datentypen

Kanonischer Repräsentant = lexikographisch kleinster

Vertreter seiner Klasse

x1 = 1  
x2 = 3  
x3 = "hallo"  
x4[1] = A  
x4[2] = B  
x4[3] = C  
x4[4] = C

<

x1 = 2  
x2 = 3  
x3 = "hallo"  
x4[1] = C  
x4[2] = A  
x4[3] = C  
x4[4] = B

x1 = 1  
x2 = 2  
x3 = "hallo"  
x4[1] = A  
x4[2] = C  
x4[3] = B  
x4[4] = C

1 → 1  
3 → 2  
2 → 3  
4 → 4

x1 = 1  
x2 = 2  
x3 = "hallo"  
x4[1] = A  
x4[2] = C  
x4[3] = B  
x4[4] = C

2 → 1  
3 → 2  
4 → 3  
1 → 4

≈ Sortieren

# Symmetrie in Datentypen

Fall 2: wie skalare Datentypen, mit zusätzlichen Operationen  
INCR(x) und DECR(x), die wie Addition/Subtraktion von 1  
modulo n funktioniert

→ Nur Rotationen (= Permutationen der Form  $a_i \rightarrow a_{(i+k) \bmod n}$ )  
generieren Symmetrien.

andere Permutationen liefern potenziell unverträgliche Werte  
für Terme der Form INCR(x)

Beispiel: s:  $x_1 = 2$

$\pi$ :  $1 \rightarrow 2$

$2 \rightarrow 1$

$3 \rightarrow 3$

$4 \rightarrow 4$

$\sigma(s)$ :  $x_1 = 1$

INCR(x<sub>1</sub>) = 2  
 $\neq \pi(3)$

INCR(x<sub>1</sub>) = 3

# Letzte Folie zu Symmetrie in Datentypen

- Man kann auch mehrere Skalar- und Rotationstypen gleichzeitig handhaben
- prinzipiell könnte man die Liste der Datentypen erweitern
- Tools nutzen meist nur skalare Typen
- Methode der kanonischen Repräsentanten ist effizient
- Der Nutzer ist dafür verantwortlich, die Datentypen geeignet zu wählen und damit die Symmetrie dem Tool zu vermitteln

# 3.5.5 Komponentensymmetrie

Basis: gerichteter, an Kanten und Knoten beschrifteter Graph

[V,E,c]

-V Knotenmenge

-E  $\subseteq V \times V$  gerichtete Kantenmenge

-c:  $V \cup E \rightarrow D$  Beschriftung (D – irgendeine Menge)

Herkunft:

z.B. Knoten = Komponenten, Kanten = Komm.-kanäle

z.B. Knoten, Kanten eines Petrinetzes, eines SDL-Systems  
einer UML-Beschreibung, eines Statecharts,.....

....

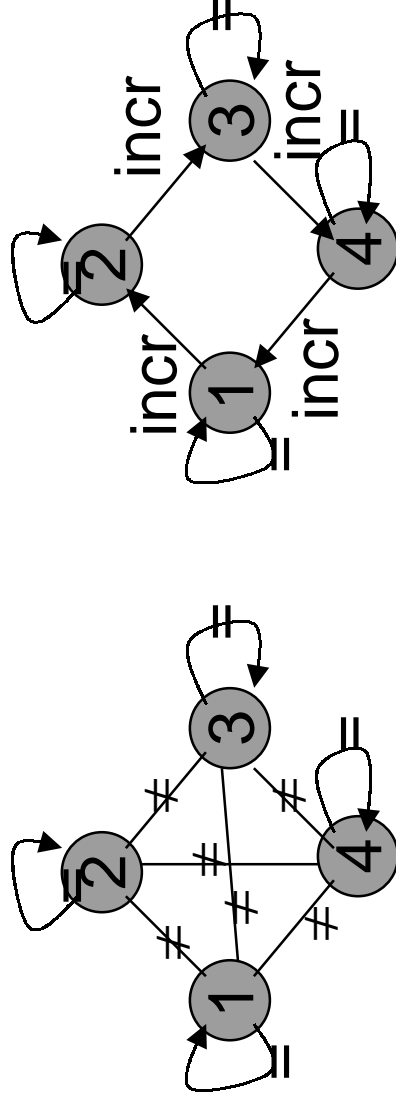
# Graphautomorphismen

Eine Permutation  $s: V \rightarrow V$  heißt Graphautomorphismus, falls für alle  $v, v'$  aus  $V$  gilt:

1.  $c(v) = c(\sigma(v))$
2. Wenn  $[v, v'] \in E$ , so  $[\sigma(v), \sigma(v')] \in E$  und  $c([v, v']) = c([\sigma(v), \sigma(v')])$

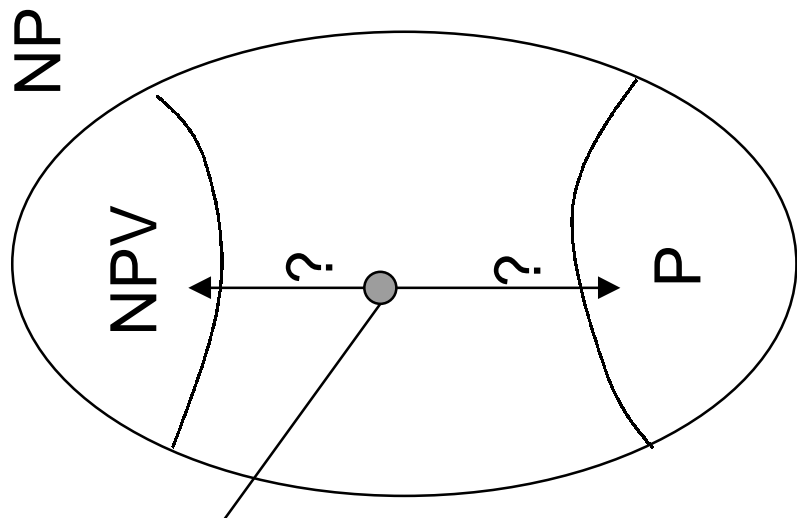
Graphautomorphismen einer graphischen Systembeschreibung induzieren Symmetrien im zug. Transitionssystem

Hinter allen Symmetrieansätzen stecken Graphautomorphismen, z.B. auch hinter Datentypsymmetrie:



# Komplexität des Automorphismenproblems

eng verwandt: Graphisomorphie



Ein Graph kann exponentiell  
viele Automorphismen haben

# Weiteres Vorgehen

1. ein bißchen Gruppentheorie, mit dem Ziel:
2. ein polynomiell großes Erzeugendensystem
3. Berechnung des Erzeugendensystems
4. Lösung des Orbit-Problems

# Gruppentheorie

$[G, o]$  ist Gruppe, wenn  $o$  assoziative Operation auf  $G$  ist, es ein neutrales Element  $e$  gibt, und jedes Element  $g$  ein Inverses  $g^{-1}$  hat.

Beispiele:

Menge der Graphautomorphismen mit Hintereinanderausführung.

Menge der ganzen Zahlen mit Addition

Menge der positiven rationalen Zahlen mit Multiplikation

# Untergruppen

$U \subseteq G$  heißt Untergruppe, wenn  $0$  nicht aus  $U$  herausführt und  $[U, 0]$  Gruppe ist.

Beispiele:

$\{1d\}$

Die durch  $k$  teilbaren ganzen Zahlen mit  $+$

diejenigen rationalen Zahlen, wo Zähler und Nenner Zweierpotenzen sind, mit Multiplikation

# Orbits

Jede Untergruppe  $U$  definiert eine Äquivalenzrelation auf  $G$ :

$$g \sim_U g' \quad \text{gdw.} \quad g \circ g'^{-1} \in U$$

$$\text{reflexiv: } g \circ g^{-1} = e \in U$$

$$\text{symmetrisch: Sei } g \circ g'^{-1} = u \in U$$

$$g' \circ g^{-1} = (g \circ g'^{-1})^{-1} = u^{-1} \in U$$

$$\text{transitiv: Sei } g \circ g'^{-1} = u \in U \text{ und } g' \circ g''^{-1} = u' \in U$$

$$\begin{aligned} g \circ g''^{-1} &= g \circ (g'^{-1} \circ g') \circ g''^{-1} \\ &= (g \circ g'^{-1}) \circ (g' \circ g''^{-1}) \\ &= u \circ u' \in U \end{aligned}$$

Klassen heißen Orbits

$g \sim_U g'$  gdw.  $g = g' \circ g'^{-1} \in U$

## Beispiele

$$G = [Z, +]$$

$$U = 3Z$$

Orbit1: 0, 3, 6, 9, 12, 15, 18, ..., -3, -6, ...

Orbit2: 1, 4, 7, 10, 13, 16, ..., -2, -5, ...

Orbit3: 2, 5, 8, 11, 14, 17, ..., -1, -4, ...

# Beziehung Orbit-Untergruppe

$$g \sim_U g' \quad \text{gdw.} \quad g \circ g'^{-1} \in U$$

Sei  $g^*$  irgendein fest gewähltes Element von  $G$   
und  $O_{g^*}$  sein Orbit

Dann gilt:

$$\begin{aligned} g \in O_{g^*}: & \quad \text{gdw.} \quad g \circ g^{*-1} \in U, \\ & \quad \text{gdw. ex. } u \in U \text{ mit } g \circ g^{*-1} = u \\ & \quad \text{gdw. ex. } u \in U \text{ mit } g = g^* \circ u \end{aligned}$$

$$\text{also: } O_{g^*} = \{ g^* \circ u \mid u \in U \}$$

# Erzeugung von Orbits

Sei  $g \in G$  und  $O_g$  der Orbit, in dem  $g$  liegt.

$$O_g = \{g \circ u \mid u \in U\}$$

Beispiel:  $G = [Z, +]$      $U = 3Z$

$$\text{Orbit1} = U = \dots, 0, \underline{3}, \underline{6}, \underline{9}, \dots = \dots, \underline{6} + -6, \underline{6} + -3, \underline{6} + 0, \underline{6} + 3, \dots$$

$$\text{Orbit2} = \dots, 1, \underline{4}, \underline{7}, \dots = \dots, \underline{4} + -3, \underline{4} + 0, \underline{4} + 3, \dots$$

$$\text{Orbit3} = \dots, 2, \underline{5}, \underline{8}, \dots = \dots, \underline{8} + -6, \underline{8} + -3, \underline{8} + 0, \dots$$

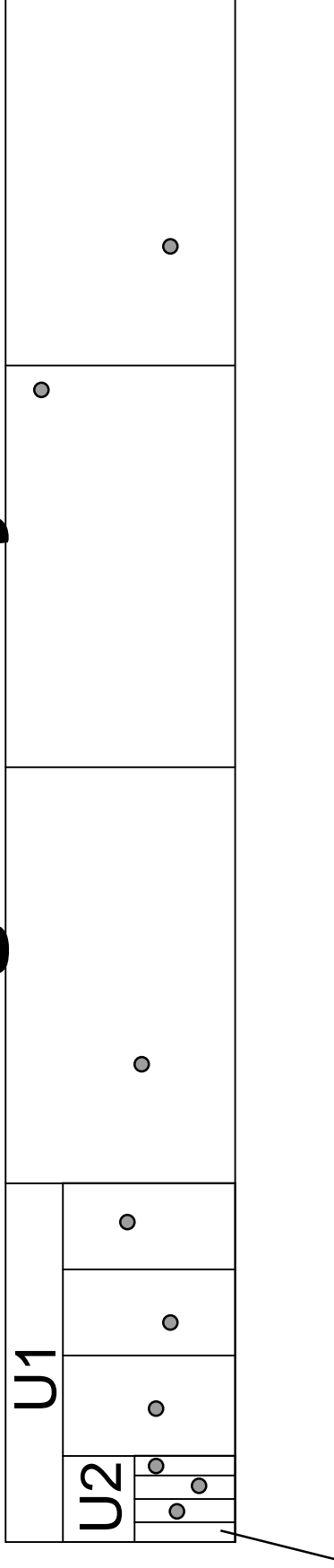
Idee für Erzeugendensystem:

Nimm ein beliebiges Element aus jedem Orbit

+ (Erzeugendensystem für)  $U$

Unser Beispiel: Aus  $3Z$ , 4 und 8 kann  $Z$  generiert werden <sup>23</sup>

# Konzept für Erzeugendensystem



$$U3 \quad U1 \supseteq U2 \supseteq U3 \supseteq \dots \supseteq U_n = \{e\}$$

Eindeutige Darstellung:

Jedes Element  $g$  von  $G$  besitzt genau eine Darstellung der Form

$g = g_1 \circ g_2 \circ \dots \circ g_n$  mit  $g_i$  aus einem der von  $U_i$  in  $U_{(i-1)}$  generierten Orbit

# Zurück zu Graphautomorphismen

[V,E,c]; Sei  $V = \{v_1, \dots, v_n\}$

$U_i = \{\pi \mid \pi \text{ ist Graphautomorphismus und } \pi(v_j) = v_j \text{ für } j \leq i\}$

Orbits  $O_{ik}$  bzgl.  $U_i$  in  $U(i-1)$ :

$\{\pi \mid \pi \text{ ist Graphautomorphismus, } \pi(v_j) = v_j \text{ für } j < i, \pi(v_i) = v_k\}$

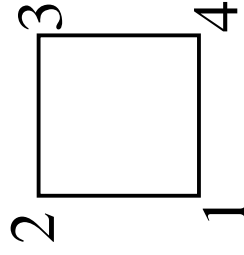
→ Erzeugendensystem:

für alle  $i \in \{1, \dots, n\}$  und  $k \in \{i+1, \dots, n\}$ :

Wenn  $O_{ik} \neq \emptyset$  dann nimm genau ein Element in das Erzeugendensystem auf (für  $O_{ii}$  immer id).

→ max.  $n(n-1)/2$  Elemente

# Beispiel



1→1	1→2	1→3	1→4	1→4
2→2	2→3	2→4	2→1	2→3
3→3	3→4	3→1	3→2	3→2
4→4	4→1	4→2	4→3	4→1
id				

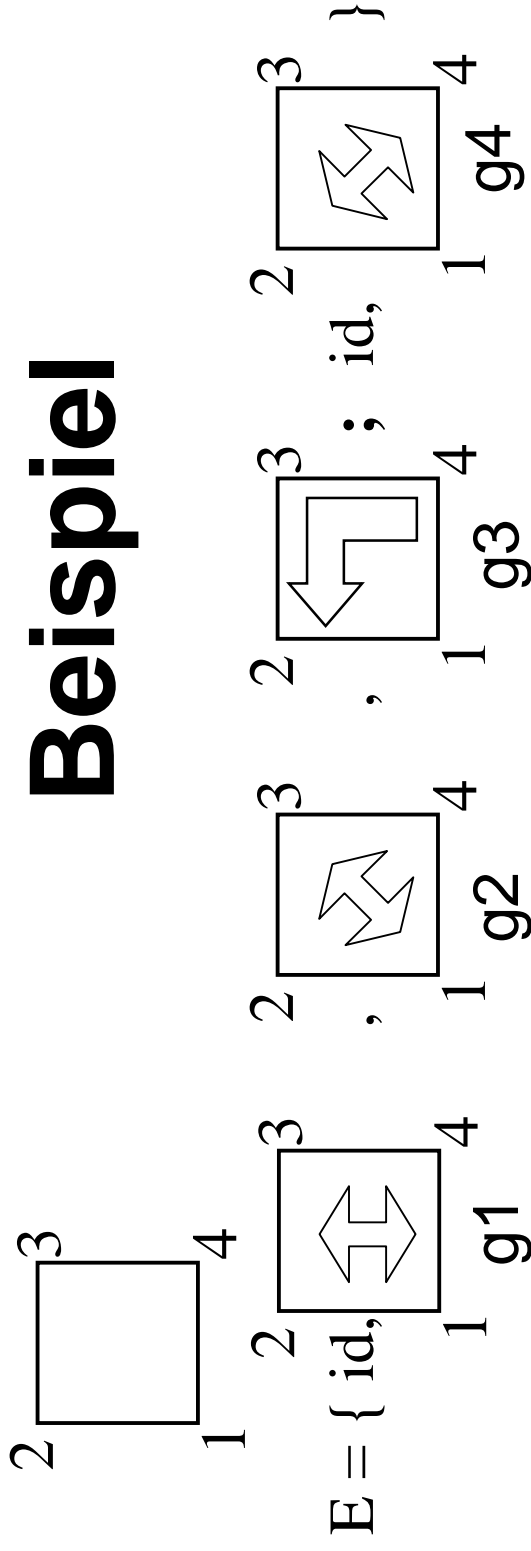
---

U1

---

U2

# Beispiel



$$\text{id} \circ \text{id} = \text{id}$$

$$\text{id} \circ g_4 = g_4$$

$$g_1 \circ \text{id} = \text{id}$$

$$g_1 \circ g_4 = g_4$$

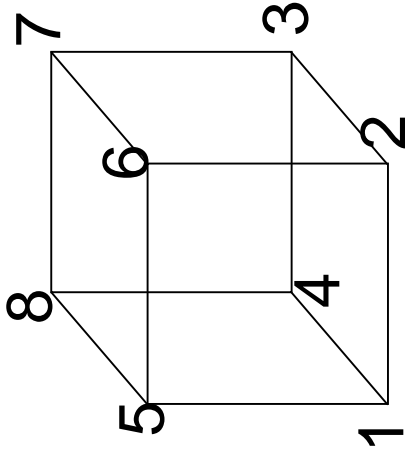
$$g_2 \circ \text{id} = g_2$$

$$g_2 \circ g_4 = g_4$$

$$g_3 \circ \text{id} = g_3$$

$$g_3 \circ g_4 = g_4$$

## Zweites Beispiel



$$g = g_1 \circ g_2 \circ g_3$$

1. Ebene:  $1 \rightarrow 1 \dots 8$
2. Ebene  $1 \rightarrow 1, 2 \rightarrow 2, 4, 5$
3. Ebene  $1 \rightarrow 1, 2 \rightarrow 2, 3 \rightarrow 3, 6$

$7 + 2 + 1 = 10$  Erzeugende für

$8 \times 3 \times 2 = 48$  Automorphismen