

**Graphersetzungssysteme**

**Annegret Habel**

**Carl v. Ossietzky Universität Oldenburg  
SS 2004**

## Inhaltsverzeichnis

<b>1</b>	<b>Einführung</b>	<b>4</b>
1.1	Inhalt . . . . .	4
1.2	Historische Entwicklung . . . . .	4
1.3	Graph-Grammatik-Ansätze . . . . .	5
1.4	Existierende Sprachen und Werkzeuge . . . . .	5
<b>2</b>	<b>Graphersetzungssysteme</b>	<b>8</b>
2.1	Graphen . . . . .	9
2.2	Graphmorphismen . . . . .	11
2.3	Graphersetzungsregel . . . . .	15
2.4	Anwendung einer Graphersetzungsregel . . . . .	16
2.5	Charakterisierung von Verklebungsdiagrammen . . . . .	20
2.6	Direkte Ableitung . . . . .	22
2.7	Die Graphgrammatik WSF . . . . .	24
2.8	Die Graphgrammatik CON . . . . .	24
2.9	Das Bibliothekssystem . . . . .	28
2.10	Das Graphersetzungssystem BIB . . . . .	29
2.11	Die Graphgrammatik BIB . . . . .	35
<b>3</b>	<b>Eigenschaften von Ableitungen</b>	<b>36</b>
3.1	Einbettbarkeit . . . . .	36
3.2	Einschränkbarkeit . . . . .	40
<b>4</b>	<b>Mächtigkeit von Graphersetzungssystemen</b>	<b>44</b>
4.1	Turingmaschine $TM$ als Graphersetzungssystem $\mathcal{G}(TM)$ . . .	45
4.2	Übersetzung von Chomsky-Grammatiken . . . . .	46
4.3	Übersetzung von Instanzen des PCP . . . . .	48
<b>5</b>	<b>Unabhängigkeit von Ableitungen</b>	<b>51</b>
<b>6</b>	<b>Parallelableitungen</b>	<b>58</b>

<b>7 Konfluenz</b>	<b>62</b>
<b>8 Graphprogramme</b>	<b>72</b>
8.1 Programs . . . . .	73
8.2 Example: programs . . . . .	73
8.3 Completeness . . . . .	73
8.4 Graph Expressions . . . . .	74
8.5 Representing abstract graphs . . . . .	75
<b>A Kategorientheorie</b>	<b>77</b>
A.1 Eigenschaften von Graphmorphismen . . . . .	87
A.2 Verklebungsbedingung . . . . .	88
A.3 Zusammenfassung . . . . .	93
 <b>Literaturverzeichnis</b>	 <b>94</b>
 <b>Index</b>	 <b>95</b>

# 1 Einführung

## 1.1 Inhalt

- Was sind Graphersetzungssysteme?
- Was können Graphersetzungssysteme?
- Welche Eigenschaften haben Graphersetzungssysteme?
- Sind Graphprogramme berechnungsvollständig?

## 1.2 Historische Entwicklung

- 1969 **Pfaltz-Rosenfeld:** Web Grammars  
1970 **Montanari:** Separable Graphs, Planar Graphs and Web Grammars  
1971 **Pratt:** Pair Grammars, Graph Languages and String-to-Graph Translations  
1972 **Pfaltz:** Web Grammars and Picture Description
- 
- 1973 **Ehrig-Pfender-Schneider:** Graph Grammars: An Algebraic Approach  
**Nagl:** Eine Präzisierung des Pfaltz/Rosenfeldschen Produktionsbegriffs bei mehrdimensionalen Grammatiken  
1975 **Rosen:** Deriving Graphs from Graphs by Applying a Production  
⋮

---

1978	Int. Workshop on Graph Grammars ...	LNCS 73, 1979
1982	2nd Int. Workshop on Graph Grammars ...	LNCS 153, 1983
1986	3rd Int. Workshop on Graph Grammars ...	LNCS 291, 1987
1990	ESPRIT Basic Research Working Group: Computing by Graph Transformation (COMPU-GRAPH)	
	4th Int. Workshop on Graph Grammars ...	LNCS 532, 1991
1992	ESPRIT Basic Research Working Group: Computing by Graph Transformation II (COMPU-GRAPH II)	
1995	5th Int. Workshop on Graph Grammars ...	LNCS 1073, 1996
1996	TMR Research Network: General Theory of Graph Transformations (GETGRATS)	
1997	ESPRIT Working Group: Applications of Graph Transformation (APPLIGRAPH)	
1998	6th Int. Workshop on Graph Grammars ...	LNCS 1764, 2000
2002	1st Int. Conference on Graph Transformation	LNCS 2505, 2002
	⋮	

### 1.3 Graph-Grammatik-Ansätze

mengentheoretischer Ansatz	[Nag73], [Nag79], [Göt88]
algebraischer Ansatz	[EPS73], [Ehr79]
logischer Ansatz	[BC87], [Cou90]
Knoten-Ersetzungs-Ansatz	[JR80], [Roz87], [ER91]
Hyperkanten-Ersetzungs-Ansatz	[HK87], [Hab92]

### 1.4 Existierende Sprachen und Werkzeuge

PLAN2D	2-dimensionale Programmiersprache auf der Basis von Graph Transformation (Dehnert/Franck/Streng 75)
PAGGED	graphischer Editor für programmierte, attributierte Graph-Grammatiken (Göttler 88)
Graph <sup>Ed</sup>	interaktives Werkzeug zum Entwickeln von Graph-Grammatiken (Himsolt 91)
Dactl	experimentelle Graphersetzungssprache (Glaubert et al. 91)
PLEXUS	Werkzeuge zur Analyse von Graph-Grammatiken (Wanke 91)
PROGRES	Spezifikations-Sprache auf der Basis von attribuierten Graph-Grammatiken (Schürr et al. 99)
AGG	Implementierung algebraischer Graphersetzung (Ermel et al. 99)
$\Gamma$	Structured Gamma (Fradet/LeMetayer 98)
Grrr	Eine Programmiersprache auf der Basis von Graphersetzung (Rodgers 98)

## Handbücher

- [Roz97] Grzegorz Rozenberg, ed. Handbook of Graph Grammars and Computing by Graph Transformation, volume 1: Foundations. World Scientific, 1997.
- [EEKR99] Hartmut Ehrig, Gregor Engels, Hans-Jörg Kreowski, Grzegorz Rozenberg, eds. Handbook of Graph Grammars and Computing by Graph Transformation, volume 2: Applications, Languages and Tools. World Scientific, 1999.
- [EKMR99] Hartmut Ehrig, Hans-Jörg Kreowski, Ugo Montanari, Grzegorz Rozenberg, eds. Handbook of Graph Grammars and Computing by Graph Transformation, volume 3: Concurrency, Parallelism, and Distribution. World Scientific, 1999.

## Tagungsbänder

- [CER79] Volker Claus, Hartmut Ehrig, Grzegorz Rozenberg, eds. Graph Grammars and Their Application to Computer Science and Biology, Lecture Notes in Computer Science 73, 1979.

- [ENR83] Hartmut Ehrig, Manfred Nagl, Grzegorz Rozenberg, eds. Graph Grammars and Their Application to Computer Science, Lecture Notes in Computer Science 153, 1983.
- [ENRR87] Hartmut Ehrig, Manfred Nagl, Grzegorz Rozenberg, Azriel Rosenfeld, eds. Graph Grammars and Their Application to Computer Science, Lecture Notes in Computer Science 291, 1987.
- [EKR91] Hartmut Ehrig, Hans-Jörg Kreowski, Grzegorz Rozenberg, eds. Graph Grammars and Their Application to Computer Science, Lecture Notes in Computer Science 532, 1991.
- [SE94] Hans Jürgen Schneider and Hartmut Ehrig, editors. Graph Transformations in Computer Science, volume 776 of Lecture Notes in Computer Science. Springer-Verlag, 1994.
- [CEER96] Janice Cuny, Hartmut Ehrig, Gregor Engels, Grzegorz Rozenberg, eds. Graph Grammars and Their Application to Computer Science, Lecture Notes in Computer Science 1073, 1996.
- [EEKR00] Hartmut Ehrig, Gregor Engels, Hans-Jörg Kreowski, and Grzegorz Rozenberg, eds. Theory and Application of Graph Transformations, Lecture Notes in Computer Science 1764. Springer-Verlag, 2000.
- [CEKR02] Andrea Corradini, Hartmut Ehrig, Hans-Jörg Kreowski, and Grzegorz Rozenberg, editors. Graph Transformation (ICGT 2002), volume 2505 of Lecture Notes in Computer Science. Springer-Verlag, 2002.

## 2 Graphersetzungssysteme

### Graphen

- komplexe Datenobjekte, die Informationen und Beziehungen repräsentieren
- anschaulich
- mathematische Gebilde

### Graphmanipulation

- durch Ersetzungsregeln  $L \Rightarrow R$  (bewirken lokale Änderung)
- zum Erzeugen von Graphen
- zum Ändern von Zuständen
- als Berechnungsprozeß
- $\vdots$

**Graphen** bestehen aus:

**Knoten:**      $\bullet$       $\circ$

**Kanten:**      $\longrightarrow$       $\dashrightarrow$

jede Kante geht von einem Knoten aus und zu einem Knoten hin

**Markierungen** von Knoten und Kanten:

$A\bullet$       $\textcircled{A}$       $\xrightarrow{B}$       $\dashrightarrow\boxed{B}$

**Warnung:** Graph ist nicht gleich Graph!

**Variationen:**



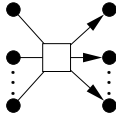
ungerichtete Kanten:



keine Mehrfachkanten:



Hyperkanten:



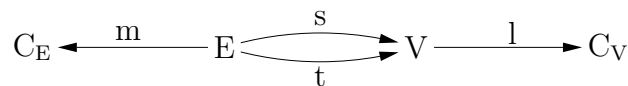
keine Markierungen (oder nur Knoten- bzw. nur Kantenmarkierungen)

## 2.1 Graphen

**Definition 1. (Graphen)** Sei  $C = \langle C_V, C_E \rangle$  ein Paar von **Markierungsalphabeten**. Ein gerichteter, markierter **Graph** über  $C$  ist ein System  $G = (V, E, s, t, l, m)$ . Dabei sind

- $V, E$  endliche Mengen von **Knoten** und **Kanten**,
- $s, t: E \rightarrow V$  Abbildungen, die jeder Kante eine **Quelle** und ein **Ziel** zuordnen,
- $l: V \rightarrow C_V$  und  $m: E \rightarrow C_E$  Abbildungen, die jedem Knoten eine **Knotenmarkierung** und jeder Kante eine **Kantenmarkierung** zuordnen.

Die Komponenten von  $G$  werden auch mit  $V_G, E_G, s_G, t_G, l_G$  und  $m_G$  bezeichnet.



**Beispiel 1.**

konkreter Graph

$$G = (V_G, E_G, s_G, t_G, l_G, m_G) \text{ mit}$$

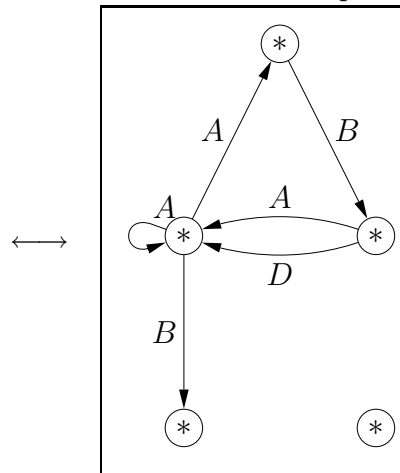
$$V_G = \{v_1, \dots, v_5\}$$

$$E_G = \{e_1, \dots, e_6\}$$

	$e_1$	$e_2$	$e_3$	$e_4$	$e_5$	$e_6$
$s_G$	$v_2$	$v_2$	$v_2$	$v_3$	$v_4$	$v_4$
$t_G$	$v_1$	$v_2$	$v_3$	$v_4$	$v_2$	$v_2$
$m_G$	$B$	$A$	$A$	$B$	$D$	$A$

$l_G(v_i) = *$  für  $i = 1, \dots, 5$   
 $*$   $\in C_V$  und  $\{A, B, D\} \subseteq C_E$

abstrakter Graph

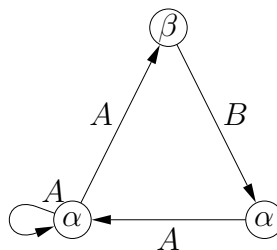


Wann haben zwei Graphen  $G$  und  $H$  die gleiche Struktur, d.h. wann sind sie als abstrakte Graphen gleich? ( $\rightarrow$  **Graphisomorphie-Problem**)

$G$	$H$																																																								
$V_G = \{1, 2, 3\}$ $E_G = \{4, 5, 6, 7\}$ <table> <tr> <th></th> <th>4</th> <th>5</th> <th>6</th> <th>7</th> </tr> <tr> <th><math>s_G</math></th> <td>1</td> <td>1</td> <td>2</td> <td>3</td> </tr> <tr> <th><math>t_G</math></th> <td>1</td> <td>2</td> <td>3</td> <td>1</td> </tr> <tr> <th><math>m_G</math></th> <td>A</td> <td>A</td> <td>B</td> <td>A</td> </tr> </table> <table> <tr> <th></th> <th>1</th> <th>2</th> <th>3</th> </tr> <tr> <th><math>l_G</math></th> <td><math>\alpha</math></td> <td><math>\beta</math></td> <td><math>\alpha</math></td> </tr> </table>		4	5	6	7	$s_G$	1	1	2	3	$t_G$	1	2	3	1	$m_G$	A	A	B	A		1	2	3	$l_G$	$\alpha$	$\beta$	$\alpha$	$V_H = \{a, b, c\}$ $E_H = \{d, e, f, g\}$ <table> <tr> <th></th> <th>d</th> <th>e</th> <th>f</th> <th>g</th> </tr> <tr> <th><math>s_H</math></th> <td>a</td> <td>b</td> <td>c</td> <td>b</td> </tr> <tr> <th><math>t_H</math></th> <td>c</td> <td>b</td> <td>b</td> <td>a</td> </tr> <tr> <th><math>m_H</math></th> <td>B</td> <td>A</td> <td>A</td> <td>A</td> </tr> </table> <table> <tr> <th></th> <th>a</th> <th>b</th> <th>c</th> </tr> <tr> <th><math>l_H</math></th> <td><math>\beta</math></td> <td><math>\alpha</math></td> <td><math>\alpha</math></td> </tr> </table>		d	e	f	g	$s_H$	a	b	c	b	$t_H$	c	b	b	a	$m_H$	B	A	A	A		a	b	c	$l_H$	$\beta$	$\alpha$	$\alpha$
	4	5	6	7																																																					
$s_G$	1	1	2	3																																																					
$t_G$	1	2	3	1																																																					
$m_G$	A	A	B	A																																																					
	1	2	3																																																						
$l_G$	$\alpha$	$\beta$	$\alpha$																																																						
	d	e	f	g																																																					
$s_H$	a	b	c	b																																																					
$t_H$	c	b	b	a																																																					
$m_H$	B	A	A	A																																																					
	a	b	c																																																						
$l_H$	$\beta$	$\alpha$	$\alpha$																																																						

$\cong?$

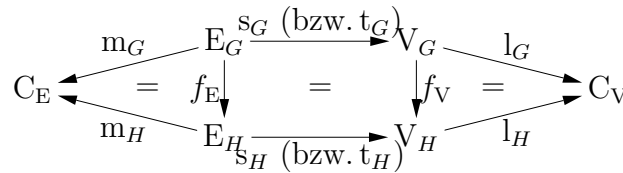
Die Graphen  $G$  und  $H$  haben die gleiche Struktur:



## 2.2 Graphmorphismen

Graphmorphismen bestehen aus struktur- und markierungserhaltenden Abbildungen zwischen Graphen:

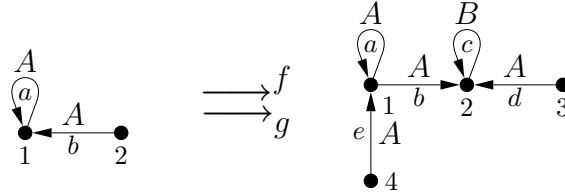
- bilden Knoten auf Knoten und Kanten auf Kanten ab,
- bewahren Quelle und Ziel von Kanten,
- bewahren Markierungen.



**Definition 2. (Graphmorphismus)** Seien  $G$  und  $H$  Graphen über  $C$ . Ein **Graphmorphismus**  $f: G \rightarrow H$  von  $G$  nach  $H$  ist ein Paar von Abbildungen  $f = \langle f_V: V_G \rightarrow V_H, f_E: E_G \rightarrow E_H \rangle$ , so daß für alle  $e \in E_G$  und alle  $v \in V_G$  gilt:

- $f_V(s_G(e)) = s_H(f_E(e))$  und  $f_V(t_G(e)) = t_H(f_E(e))$   
(Bewahrung von Quelle und Ziel)
- $l_G(v) = l_H(f_V(v))$  und  $m_G(e) = m_H(f_E(e))$   
(Bewahrung von Markierungen)

Ein Graphmorphismus  $f = \langle f_V, f_E \rangle$  heißt **injektiv (surjektiv, bijektiv)**, wenn  $f_V$  und  $f_E$  injektiv (surjektiv, bijektiv) sind. Ein bijektiver Graphmorphismus  $f: G \rightarrow H$  ist ein **Isomorphismus**. In diesem Fall heißen  $G$  und  $H$  **isomorph**, in Zeichen  $G \cong H$ . Ein **abstrakter Graph**  $[G]$  ist die Isomorphieklasse eines Graphen  $G$ :  $[G] = \{G' \mid G \cong G'\}$ .

**Beispiel 2. (Graphmorphismen)**

- $f = \left( f_V \left\{ \begin{array}{l} 1 \mapsto 1 \\ 2 \mapsto 4 \end{array} \right\}, f_E \left\{ \begin{array}{l} a \mapsto a \\ b \mapsto e \end{array} \right\} \right)$   
 $\underline{f_V(s_G(a))} = f_V(1) = 1 = s_H(a) = \underline{s_H(f_E(a))}$   
 $\dots$
- $g = \left( g_V \left\{ \begin{array}{l} 1 \mapsto 1 \\ 2 \mapsto 1 \end{array} \right\}, g_E \left\{ \begin{array}{l} a \mapsto a \\ b \mapsto a \end{array} \right\} \right)$   
 $\underline{g_V(s_G(b))} = g_V(2) = 1 = s_H(a) = \underline{s_H(g_E(b))}$   
 $\dots$

**Definition 3. (Teilgraph)** Seien  $G$  und  $H$  Graphen.  $G$  ist ein **Teilgraph** von  $H$ , in Zeichen  $G \subseteq H$ , wenn  $V_G \subseteq V_H$ ,  $E_G \subseteq E_H$  und für alle  $e \in E_G$  und alle  $v \in V_G$  gilt:  $s_G(e) = s_H(e)$ ,  $t_G(e) = t_H(e)$ ,  $l_G(v) = l_H(v)$  und  $m_G(e) = m_H(e)$ .

**Lemma 1. (Teilgraphen)** Sei  $H$  ein Graph und seien  $V_G \subseteq V_H$  und  $E_G \subseteq E_H$  Teilmengen von Knoten und Kanten mit der Eigenschaft:

$$s_H(e), t_H(e) \in V_G \text{ für alle } e \in E_G.$$

Seien  $s_G$ ,  $t_G$ ,  $l_G$  und  $m_G$  die Restriktionen der entsprechenden Abbildungen aus  $H$ . Dann ist  $G = (V_G, E_G, s_G, t_G, l_G, m_G)$  ein Teilgraph von  $H$ .

**Beweis.** Nach Voraussetzung ist  $s_H(e), t_H(e) \in V_G$  für alle  $e \in E_G$ . Folglich definieren die Restriktionen der Abbildungen aus  $H$  Abbildungen  $s_G: E_G \rightarrow V_G$ ,  $t_G: E_G \rightarrow V_G$ ,  $l_G: V_G \rightarrow C_V$  und  $m_G: E_G \rightarrow C_E$  mit  $s_G(e) = s_H(e)$ ,  $t_G(e) = t_H(e)$ ,  $l_G(v) = l_H(v)$  und  $m_G(e) = m_H(e)$  für alle  $e \in E_G$  und alle  $v \in V_G$ . Somit ist  $G$  ein Teilgraph von  $H$ .  $\square$

**Bemerkung.** Sei  $G$  ein Graph. Dann ist  $\text{id}_G = (\text{id}_V, \text{id}_E): G \rightarrow G$  mit  $\text{id}_V(v) = v$  und  $\text{id}_E(e) = e$  für alle  $v \in V_G$  und alle  $e \in E_G$  ein Graphomorphismus.

**Beweis.** Siehe Übung □

**Bemerkung.** Sei  $G \subseteq H$ . Dann ist  $\text{inc} = (\text{inc}_V, \text{inc}_E): G \rightarrow H$  mit  $\text{inc}_V(v) = v$  und  $\text{inc}_E(e) = e$  für alle  $v \in V_G$  und alle  $e \in E_G$  ein Graphomorphismus.

**Beweis.** Für alle  $e \in E_G$  und alle  $v \in V_G$  gilt:

- $\text{inc}_V(s_G(e)) = s_G(e) = s_H(e) = s_H(\text{inc}_E(e))$
- $\text{inc}_V(t_G(e)) = t_G(e) = t_H(e) = t_H(\text{inc}_E(e))$
- $l_G(v) = l_H(v) = l_H(\text{inc}_V(v))$
- $m_G(e) = m_H(e) = m_H(\text{inc}_V(e))$

Folglich ist  $\text{inc}$  ein Graphomorphismus. Nach Definition ist  $\text{inc}$  auch injektiv. □

**Lemma 2. (Komposition)** Seien  $f: G \rightarrow H$  und  $g: H \rightarrow I$  Graphomorphismen. Dann ist  $g \circ f = (g_V \circ f_V, g_E \circ f_E): G \rightarrow I$  ein Graphomorphismus.

**Beweis.** Siehe Übung □

**Lemma 3. (Eigenschaften der Komposition)**

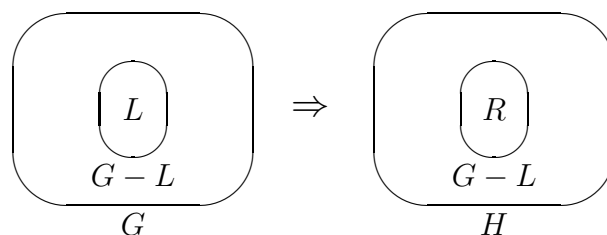
1. **Assoziativität:**  $h \circ (g \circ f) = (h \circ g) \circ f$   
für alle Graphomorphismen  $f: G \rightarrow H$ ,  $g: H \rightarrow I$ ,  $h: I \rightarrow J$ .
2. **Identität:**  $f \circ \text{id}_G = f$  und  $\text{id}_G \circ g = g$   
für alle Graphomorphismen  $f: G \rightarrow H$ ,  $g: I \rightarrow G$ .

**Beweis.** Siehe Übung □

**Bemerkung.** Die Graphen und Graphomorphismen bilden zusammen mit der Komposition eine Kategorie **GRAPHS**.

### Idee der regelbasierten Graphmanipulation

- Regel  $L \Rightarrow R$  beschreibt lokale Änderung
- Anwendung von  $L \Rightarrow R$  durch Ersetzung von  $L$  durch  $R$  ist in beliebiger Umgebung



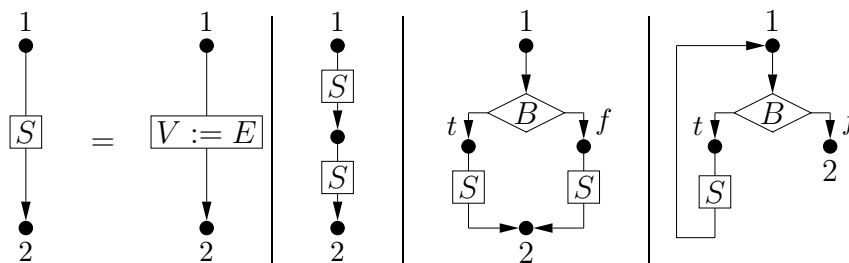
### Probleme:

- In welcher Form darf  $L$  in  $G$  vorkommen?  
(Teilgraph? Teilgraph bis auf Isomorphie? ...)
- Was passiert mit Kanten in  $G - L$ , die Knoten in  $L$  berühren?
- Wie wird  $R$  mit  $G - L$  verbunden?

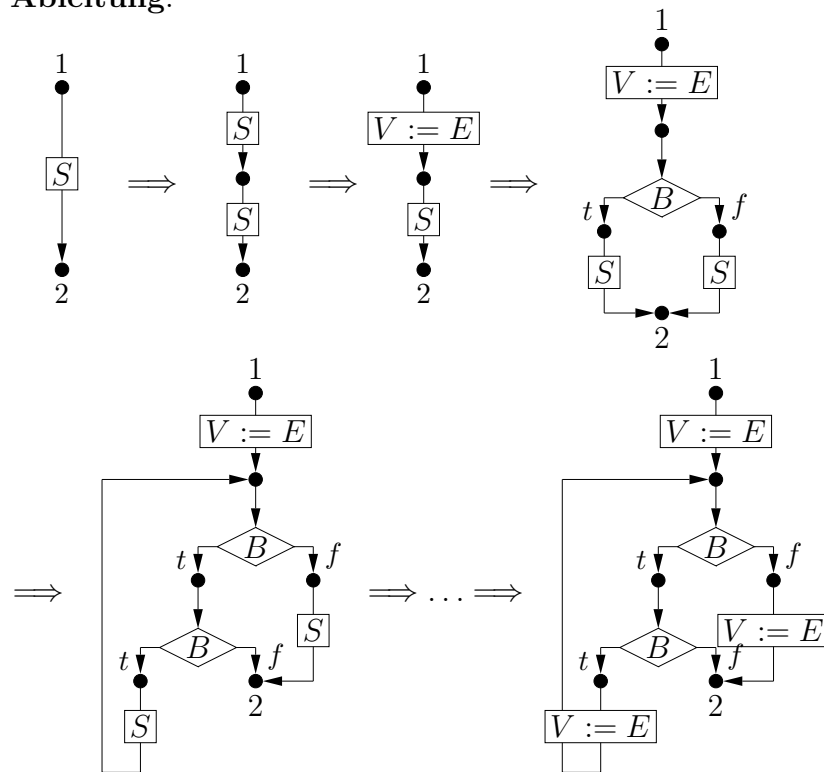
**Beispiel 3. (Erzeugung von Flußdiagrammen)** Syntax von Anweisungen (in BNF):

$$S ::= \begin{array}{c} V := E \\ \text{(Zuweisung)} \end{array} \mid \begin{array}{c} S ; S \\ \text{(Sequenz)} \end{array} \mid \begin{array}{c} \text{if } B \text{ then } S \text{ else } S \\ \text{(Alternative)} \end{array} \mid \begin{array}{c} \text{while } B \text{ do } S \\ \text{(Schleife)} \end{array}$$

Graphersetzungsgesetze:



Ableitung:



## 2.3 Graphersetzungsregel

**Definition 4. (Graphersetzungsregel)** Eine **Graphersetzungsregel** (**Regel**) über  $C$  hat die Form  $r = (L \supseteq K \subseteq R)$ , wobei  $L$ ,  $K$ , und  $R$  Graphen über  $C$  sind.  $L$  heißt **linke Seite**,  $R$  **rechte Seite** und  $K$  der **Klebegraph** von  $r$ .

**Idee der Anwendung:**

- Lösche ein Vorkommen von  $L - K$
- Füge  $R - K$  hinzu

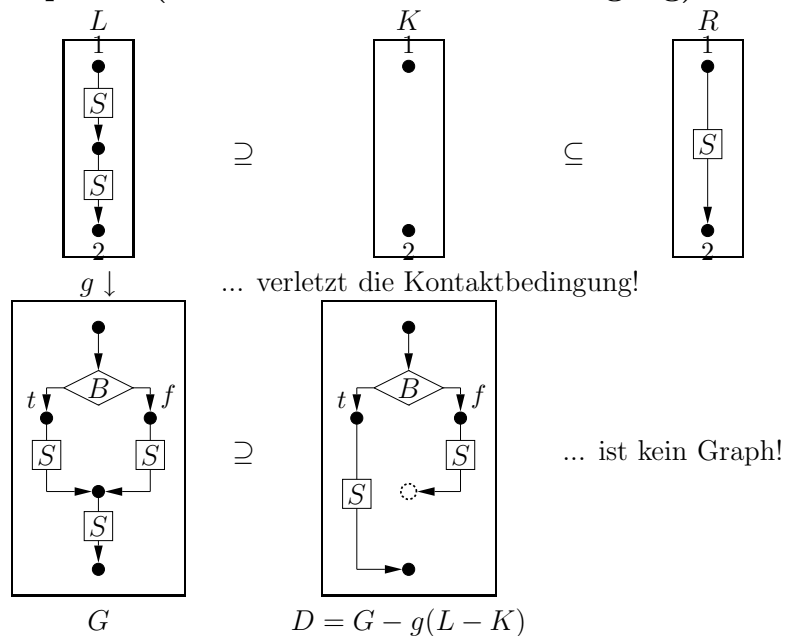
$K$  ist die "Schnittstelle"

## 2.4 Anwendung einer Graphersetzungsregel

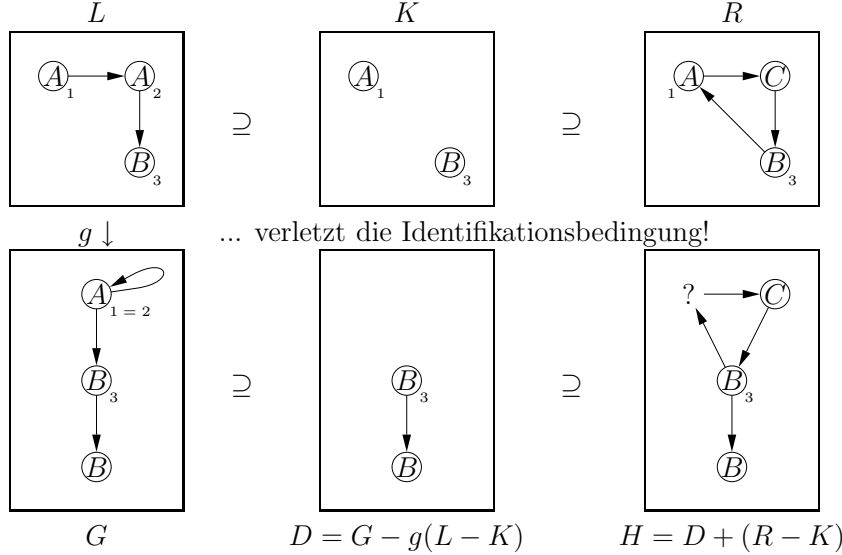
Anwendung von  $r = (L \supseteq K \subseteq R)$  auf  $G$ :

- (1) Wähle ein Vorkommen von  $L$  in  $G$ , d.h. einen Graphmorphismus  $g: L \rightarrow G$ .
- (2) Überprüfe die Kontakt- und Identifikationsbedingung.
- (3) Lösche  $g(L - K)$ , d.h. alle Kanten in  $g_E(E_L - E_K)$  und alle Knoten in  $g_V(V_L - V_K)$ . Zwischenergebnis:  $D = G - g(L - K)$ .
- (4) Füge  $R - K$  hinzu, d.h. alle Knoten in  $V_R - V_K$  und alle Kanten in  $E_R - E_K$ .  
Ergebnis:  $H = D + (R - K)$ .

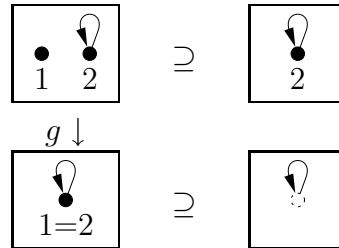
**Beispiel 4. (Löschen ohne Kontaktbedingung)**





**Beispiel 5. (Löschen ohne Identifikationsbedingung)****Problem beim Löschen**

Das Löschen von  $g(L - K)$  in  $G$  liefert i.a. keinen Graphen, d.h. die Knoten- und Kantenmengen  $V_D = V_G - g_V(V_L - V_K)$  und  $E_D = E_G - g_E(E_L - E_K)$  definieren nicht immer einen Teilgraphen  $D$  von  $G$ . Dies passiert, wenn Kanten außerhalb des Bildes von  $L$  Knoten berühren, die gelöscht werden, d.h. wenn Kanten aus  $G - g(L)$  Knoten aus  $g(L - K)$  berühren oder wenn ein Knoten aus  $L - K$  mit dem Quell- oder Zielknoten einer Klebekante identifiziert wird:



**Lemma 4. (Löschen I)** Seien  $L$  und  $K$  Graphen mit  $K \subseteq L$ , und sei  $g: L \rightarrow G$  ein Graphomorphismus, der die folgende **Kontaktbedingung** erfüllt:

Für alle  $e \in E_G - g_E(E_L)$  gilt:  $s_G(e), t_G(e) \notin g_V(V_L - V_K)$ .

Dann ist  $D = (V_D, E_D, s_D, t_D, l_D, m_D)$  mit

$$\begin{array}{rcl}
V_D & = & V_G - (g_V(V_L) - g_V(V_K)) \\
E_D & = & E_G - (g_E(E_L) - g_E(E_K)) \\
s_D(e) & = & s_G(e) \\
t_D(e) & = & t_G(e) \\
l_D(v) & = & l_G(v) \\
m_D(e) & = & m_G(e)
\end{array} \left. \vphantom{\begin{array}{rcl} V_D \\ E_D \\ s_D(e) \\ t_D(e) \\ l_D(v) \\ m_D(e) \end{array}} \right\} \text{ für alle } e \in E_D \text{ und alle } v \in V_D$$

ein Teilgraph von  $G$ .

**Beweis.** Zu zeigen: Für alle  $e \in E_D$ :  $s_G(e), t_G(e) \in V_D$ .

**Fall 1.** Sei  $e \in E_G - g_E(E_L)$ . Dann gilt wegen der Kontaktbedingung  $s_G(e), t_G(e) \in V_G - (g_V(V_L) - g_V(V_K)) = V_D$ .

**Fall 2.** Sei  $e \in g_E(E_K)$ . Dann existiert ein  $e' \in E_K$  mit  $g_E(e') = e$  und es gilt  $s_G(e) = s_G(g_E(e')) = g_V(s_L(e')) = g_V(s_K(e')) \in g_V(V_K) \subseteq V_D$  und entsprechend  $t_G(e) \in V_D$ .  $\square$

**Lemma 5. (Löschen II)** Seien  $L$  und  $K$  Graphen mit  $K \subseteq L$ , und  $g: L \rightarrow G$  ein Graphomorphismus, der die Kontaktbedingung und die **Identifikationsbedingung** erfüllt:

Für alle  $x, y \in L$  gilt:  $g(x) = g(y)$  impliziert  $x = y$  oder  $x, y \in K$ .

Dann ist  $D = (V_D, E_D, s_D, t_D, l_D, m_D)$  mit

$$\begin{array}{rcl}
V_D & = & V_G - g_V(V_L - V_K) \\
E_D & = & E_G - g_E(E_L - E_K) \\
s_D(e) & = & s_G(e) \\
t_D(e) & = & t_G(e) \\
l_D(v) & = & l_G(v) \\
m_D(e) & = & m_G(e)
\end{array} \left. \vphantom{\begin{array}{rcl} V_D \\ E_D \\ s_D(e) \\ t_D(e) \\ l_D(v) \\ m_D(e) \end{array}} \right\} \text{ für alle } e \in E_D \text{ und alle } v \in V_D$$

ein Teilgraph von  $G$  und  $d: K \rightarrow D$  mit  $d_V(v) = g_V(v)$  für alle  $v \in V_K$ ,  $d_E(e) = g_E(e)$  für alle  $e \in E_K$  ein Graphomorphismus von  $K$  nach  $D$ .

**Beweis.** Zu zeigen: Für alle  $e \in E_D$ :  $s_G(e), t_G(e) \in V_D$ .

**Fall 1:**  $e \in E_G - g_E(E_L)$ . Dann ist wegen der Kontaktbedingung  $s_G(e), t_G(e) \in V_G - (g_V(V_L) - g_V(V_K))$ . Ist  $s_G(e) \in V_G - g_V(V_L)$ , so ist  $s_G(e) \in V_G - g_V(V_L - V_K)$ . Ist  $s_G(e) \in g_V(V_K)$ , so existiert ein  $x' \in V_K$  mit  $g_V(x') = s_G(e)$ .

Angenommen  $s_G(e) \in g_V(V_L - V_K)$ . Dann existiert  $x'' \in V_L - V_K$ :  $g_V(x'') = s_G(e)$ . Wegen der Identifikationsbedingung ist  $x' = x''$  oder  $x', x'' \in V_K$ . Widerspruch. Also ist  $s_G(e) \in V_G - g_V(V_L - V_K)$ .

**Fall 2:**  $e \in g_E(E_L) - g_E(E_L - E_K)$ . Dann ist  $e \in g_E(E_K)$  und es existiert ein  $e' \in E_K$ :  $g_E(e') = e$ . Folglich ist  $s_G(e) = s_G(g_E(e')) = g_V(s_L(e')) = g_V(s_K(e')) \subseteq g_V(V_K)$ , d.h., es existiert  $x' \in V_K$ :  $g_V(x') = s_G(e)$ . Angenommen  $s_G(e) \in g_V(V_L - V_K)$ . Dann existiert ein  $x'' \in V_L - V_K$ :  $g_V(x'') = s_G(e)$ . Wegen der Identifikationsbedingung ist  $x' = x''$  oder  $x', x'' \in V_K$ . Widerspruch. Also ist  $s_G(e) \in V_G - g_V(V_L - V_K)$ .

$d: K \rightarrow D$  ist ein Graphmorphismus:  $s_D(d_E(e)) = s_G(d_E(e)) = s_G(g_E(e)) = g_V(s_L(e)) = g_V(s_K(e)) = d_V(s_K(e))$  Entsprechend gilt  $t_D(d_E(e)) = d_V(t_K(e))$ ,  $l_D(d_V(v)) = l_K(v)$  und  $m_D(d_E(e)) = m_K(e)$ .  $\square$

### Verkleben (Hinzufügen)

Seien  $K$  und  $R$  Graphen mit  $K \subseteq R$ , und sei  $d: K \rightarrow D$  ein Graphmorphismus. Dann ist der wie folgt konstruierte Graph  $H = (V_H, E_H, s_H, t_H, l_H, m_H)$  die **Verklebung** von  $D$  und  $R$  gemäß  $d$ :

$$\begin{aligned} V_H &= V_D + (V_R - V_K) \\ E_H &= E_D + (E_R - E_K) \\ s_H: E_H &\rightarrow V_H \text{ mit } s_H(e) = \begin{cases} s_D(e) & \text{für } e \in E_D \\ d_V(s_R(e)) & \text{für } e \in E_R - E_K \text{ mit } s_R(e) \in V_K \\ s_R(e) & \text{sonst} \end{cases} \\ t_H: E_H &\rightarrow V_H \text{ analog zu } s_H \\ l_H: V_H &\rightarrow C_V \text{ mit } l_H(v) = \begin{cases} l_D(v) & \text{für } v \in V_D \\ l_R(v) & \text{sonst} \end{cases} \\ m_H: E_H &\rightarrow C_E \text{ analog zu } l_H \end{aligned}$$

**Lemma 6. (Eigenschaften der Verklebung)** Die Verklebung  $H$  von  $D$  und  $R$  gemäß  $d$  hat folgende Eigenschaften:

- (1)  $D \subseteq H$
- (2)  $h: R \rightarrow H$  mit  $h(x) = \begin{cases} x & \text{für } x \in R - K \\ d(x) & \text{sonst} \end{cases}$  ist ein Graphmorphismus.
- (3)  $d: K \rightarrow D$  ist die Einschränkung von  $h: R \rightarrow H$  auf  $K$  und  $D$ .

$$\begin{array}{ccc} K & \subseteq & R \\ d \downarrow & & \downarrow h \\ D & \subseteq & H \end{array}$$

**Beweis. Zu zeigen:**  $D$  ist ein Teilgraph von  $H$ . Nach Definition von  $H$  gilt:  $V_D \subseteq V_H$ ,  $E_D \subseteq E_H$ ,  $s_D(e) = s_H(e)$ ,  $t_D(e) = t_H(e)$ ,  $l_D(v) = l_H(v)$ ,  $m_D(e) = m_H(e)$  für alle  $e \in E_D$  und alle  $v \in V_D$ . Folglich ist  $D$  ein Teilgraph von  $H$ .

**Zu zeigen:**  $h: R \rightarrow H$  ist ein Graphmorphismus.

**Beh.**  $h_V(s_R(e)) = s_H(h_E(e))$  für alle  $e \in E_R$ .

**Fall 1:**  $e \in E_K$ . Dann ist  $s_R(e) = s_K(e) \in V_K$  und  $h_V(s_R(e)) = d_V(s_K(e)) = s_D(d_E(e)) = s_H(d_E(e)) = s_H(h_E(e))$ .

**Fall 2:**  $e \in E_R - E_K$  und  $s_R(e) \in V_K$ .

Dann ist  $h_V(s_R(e)) = d_V(s_R(e)) = s_H(e) = s_H(h_E(e))$ .

**Fall 3:**  $e \in E_R - E_K$  und  $s_R(e) \in V_R - V_K$ .

Dann ist  $h_V(s_R(e)) = s_R(e) = s_H(e) = s_H(h_E(e))$ . Analog folgen  $h_V(t_R(e)) = t_H(h_E(e))$ ,  $l_R(v) = l_H(h_V(v))$  und  $m_R(e) = m_H(h_E(e))$  für alle  $e \in E_R$  und alle  $v \in V_R$ .

**Zu zeigen:**  $d: K \rightarrow D$  ist eine Einschränkung von  $h: R \rightarrow D$  auf  $K$  und  $D$ . Nach Definition von  $h$  ist  $d_V(v) = h_V(v)$  und  $d_E(e) = h_E(e)$  für alle  $v \in V_K$  und alle  $e \in E_K$ .  $\square$

## 2.5 Charakterisierung von Verklebungsdiagrammen

**Lemma 7. (Universelle Charakterisierung von Verklebungsdiagrammen)** Seien  $K$  und  $R$  und  $D$  Graphen und  $b: K \rightarrow R$  und  $d: K \rightarrow D$  Graphmorphismen. Speziell sei  $b: K \rightarrow R$  Inklusion.

$$\begin{array}{ccc} K & \xrightarrow{b} & R \\ d \downarrow & (1) & \downarrow h \\ D & \xrightarrow{c} & H \end{array}$$

(1) Verklebungs-Diagramm  $\iff$  (1) ist Pushout-Diagramm.

**Beweis.** “ $\Rightarrow$ ”: Sei (1) Verklebungs-Diagramm. Dann ist  $H$  Verklebung von  $R$  und  $D$  in  $K$ ,  $c: D \rightarrow H$  Inklusion und  $h: R \rightarrow H$  der Graphmorphismus mit

$$h(x) = \begin{cases} d(x) & \text{falls } x \in K \\ x & \text{sonst} \end{cases}$$

Dann ist  $h \circ b = c \circ d$  (nach Definition von  $c$  und  $d$ ). Sei  $H'$  ein Graph und  $h': R \rightarrow H'$  und  $c': D \rightarrow H'$  Graphmorphismen mit  $h' \circ b = c' \circ d$ . Sei weiter

$$u(x) = \begin{cases} c'(x) & \text{falls } x \in D \\ h'(x) & \text{sonst, d.h. } x \in R - K \end{cases}$$

Dann gilt  $u \circ c = c'$  (da  $c$  Inklusion) und  $u \circ h = h'$  da für  $x \in R - K$ :

$$u(h(x)) \stackrel{\text{Def. } h}{=} u(x) \stackrel{\text{Def. } u}{=} h'(x) \text{ und für } x \in K:$$

$$u(h(x)) \stackrel{\text{Def. } h}{=} u(d(x)) \stackrel{\text{Def. } u}{=} c'(d(x)) \stackrel{\text{Komm.}}{=} h'(b(x)) \stackrel{b \text{ Inkl.}}{=} h'(x).$$

Jedes  $u: H \rightarrow H'$  mit  $u \circ c = c'$  und  $u \circ h = h'$  muß so definiert sein. Folglich ist  $u$  eindeutig. Bleibt zu zeigen:  $u: H \rightarrow H'$  ist ein Graphmorphismus.

**Behauptung 1:**  $u_V(s_H(e)) = s_{H'}(u_E(e))$  für alle  $e \in E_H$ .

**Fall 1:**  $e \in E_D$ .

$$u_V(s_H(e)) \stackrel{\text{Def. } s_H}{=} u_V(s_D(e)) \stackrel{\text{Def. } u}{=} c'_V(s_D(e)) \stackrel{c' \text{ Morph.}}{=} s_{H'}(c'_E(e)) \stackrel{\text{Def. } u}{=} s_{H'}(u_E(e)).$$

**Fall 2:**  $e \in E_R - E_K$  und  $s_H(e) \in V_R - V_K$ .

$$u_V(s_H(e)) \stackrel{\text{Def. } s_H}{=} u_V(s_R(e)) \stackrel{\text{Def. } u}{=} h'_V(s_R(e)) \stackrel{h' \text{ Morph.}}{=} s_{H'}(h'_E(e)) \stackrel{\text{Def. } u}{=} s_{H'}(u_E(e)).$$

**Fall 3:**  $e \in E_R - E_K$  und  $s_H(e) \in V_D$ . Dann existiert ein  $v \in V_K$  mit  $s_R(e) = v$  und  $s_H(e) = d_V(v)$ .

$$u_V(s_H(e)) \stackrel{\text{Voraus.}}{=} u_V(d_V(v)) \stackrel{\text{Def. } u}{=} c'_V(d_V(v)) \stackrel{\text{Komm.}}{=} h'_V(b_V(v)) \stackrel{b \text{ Inkl.}}{=} h'_V(s_R(e)) \stackrel{h' \text{ Morph.}}{=} s_{H'}(h'_E(e)) \stackrel{\text{Def. } u}{=} s_{H'}(u_E(e)).$$

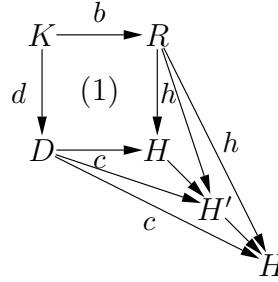
**Behauptung 2:**  $u_V(t_H(e)) = t_{H'}(u_E(e))$  für alle  $e \in E_H$ . Analog zu Behauptung 1.

**Behauptung 3:**  $l_H(v) = l_{H'}(u_V(v))$  für alle  $v \in V_H$ .

$$\text{Für } v \in V_D: l_H(v) \stackrel{\text{Def. } l_H}{=} l_D(v) \stackrel{c' \text{ Morph.}}{=} l_{H'}(c'_V(v)) \stackrel{\text{Def. } u}{=} l_{H'}(u_V(v)).$$

$$\text{Für } v \in V_R - V_K: l_H(v) \stackrel{\text{Def. } l_H}{=} l_R(v) \stackrel{h' \text{ Morph.}}{=} l_{H'}(h'_V(v)) \stackrel{\text{Def. } u}{=} l_{H'}(u_V(v)).$$

**Behauptung 4:**  $m_H(e) = m_{H'}(u_V(e))$  für alle  $e \in E_H$ . Analog zu Behauptung 3.



“ $\Leftarrow$ ”: Sei (1) ein Pushout-Diagramm. Sei  $H'$  die Verklebung von  $R$  und  $D$  in  $K$  und  $h': R \rightarrow H'$  und  $c': D \rightarrow H'$  die dazugehörigen Graphmorphismen.

**Behauptung:**  $H$  und  $H'$  sind bis auf Isomorphie gleich.

**Beweis:** Da (1) ein Pushout-Diagramm ist existiert eindeutig ein Graphmorphismus

$$u: H \rightarrow H' \text{ mit } u \circ h = h' \text{ und } u \circ c = c'.$$

Nach Teil 1 des Beweises existiert eindeutig ein Graphmorphismus

$$u': H' \rightarrow H \text{ mit } u' \circ h' = h \text{ und } u' \circ c' = c.$$

Folglich ist  $u' \circ u: H \rightarrow H$  mit  $(u' \circ u) \circ h = u' \circ h' = h$  und  $(u' \circ u) \circ c = u' \circ c' = c$  ein Graphmorphismus von  $H$  nach  $H$ . Andererseits ist  $\text{id}_H: H \rightarrow H$  mit  $\text{id}_H \circ h = h$  und  $\text{id}_H \circ c = c$  ein Graphmorphismus von  $H$  nach  $H$ . Wegen der Eindeutigkeit folgt  $u' \circ u = \text{id}_H$ . Analog folgt  $u \circ u' = \text{id}_{H'}$ . Also ist  $u': H' \rightarrow H$  ein Isomorphismus und  $H$  und  $H'$  sind isomorph.  $\square$

## 2.6 Direkte Ableitung

**Definition 5. (direkte Ableitung)** Sei  $G$  ein Graph,  $r = (L \supseteq K \subseteq R)$  eine Regel,  $g: L \rightarrow G$  ein Graphmorphismus, der die Kontakt- und Identifikationsbedingung erfüllt, und  $M$  isomorph zu dem Ergebnis der Anwendung von  $r$  auf  $G$ :

$$\begin{array}{ccccc} L & \supseteq & K & \subseteq & R \\ g \downarrow & & d \downarrow & & \downarrow h \\ G & \supseteq & D & \subseteq & H \xrightarrow{i} M \end{array}$$

Dann heit  $G \Rightarrow_{r,g} M$  **direkte Ableitung** von  $G$  nach  $M$  bezglich  $r$  und  $g$ . Hierfr schreiben wir auch  $G \Rightarrow_r M$  oder kurz  $G \Rightarrow M$ . Wir schreiben  $G \Rightarrow_{\mathcal{R}} M$  falls  $r \in \mathcal{R}$  ist.  $G \Rightarrow^* M$  heit **Ableitung** von  $G$  nach  $M$ , wenn  $G \cong M$  oder wenn es eine Folge direkter Ableitungen der Form  $G = G_0 \Rightarrow_{r_1, g_1} \dots \Rightarrow_{r_n, g_n} G_n = M$  gibt. Wir schreiben  $G \Rightarrow_{\mathcal{R}}^* M$  falls  $r_1, \dots, r_n \in \mathcal{R}$ .

**Definition 6. (Graphersetzungssystem)** Ein **Graphersetzungssystem** ist ein System  $\mathcal{G} = \langle C, \mathcal{R} \rangle$ , wobei  $C$  ein Paar von Markierungsalphabeten und  $\mathcal{R}$  eine Menge von Graphersetzungsregeln ber  $C$  ist.

**Definition 7. (Graphgrammatik und erzeugte Sprache)** Eine **Graphgrammatik** ist ein System  $\mathcal{G} = \langle C, \mathcal{R}, N, S \rangle$ , wobei  $\langle C, \mathcal{R} \rangle$  ein Graphersetzungssystem,  $N = \langle N_V, N_E \rangle$  ein Paar von nichtterminalen Markierungsalphabeten mit  $N_V \subseteq C_V$  und  $N_E \subseteq C_E$  und  $S$  ein Graph ber  $C$  ist. Die von  $\mathcal{G}$  **erzeugte Graphsprache**  $L(\mathcal{G})$  besteht aus allen Graphen  $G$ , die mit Hilfe der Regeln in  $\mathcal{R}$  aus  $S$  ableitbar sind und in denen keine nichtterminal markierten Knoten oder Kanten vorkommen:

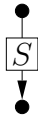
$$L(\mathcal{G}) = \{G \in \mathcal{G}_{C-N} \mid S \Rightarrow_{\mathcal{R}}^* G\}.$$

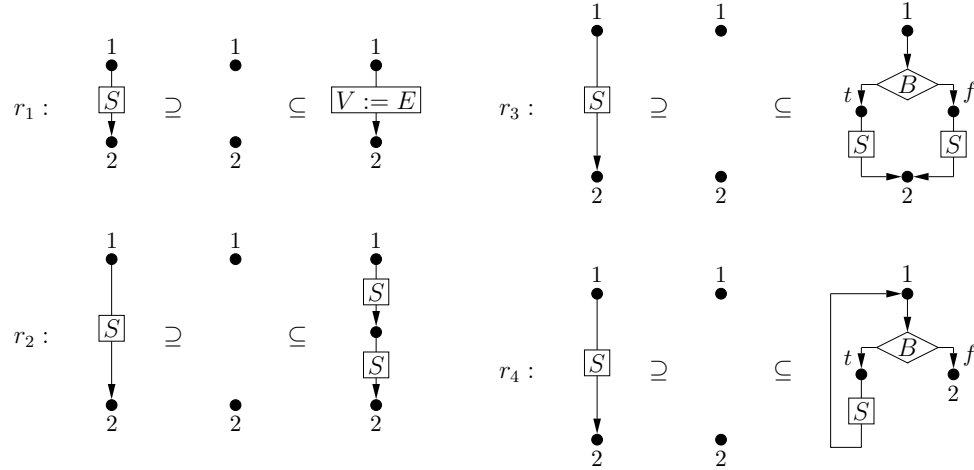
Dabei bezeichnet  $C - N$  das Paar  $\langle C_V - N_V, C_E - N_E \rangle$ .

## 2.7 Die Graphgrammatik WSF

**Beispiel 6. (Graphgrammatik)** Sei  $WSF = \langle C, \mathcal{R}, N, S_{WSF} \rangle$  die Graphgrammatik zur Erzeugung wohlstrukturierter Flußdiagramme mit

- $C_V = \{B, \square\}$  und  $C_E = \{S, V := E, t, f, \square\}$
- $N_V = \emptyset$  und  $N_E = \{S\}$

- $S_{WSF} =$ 




## 2.8 Die Graphgrammatik CON

**Definition 8. (zusammenhängender Graphen)** Ein Graph heißt **zusammenhängend**, wenn je zwei Knoten verbunden sind.

Zwei Knoten  $v_0$  und  $v_n$  sind **verbunden**, wenn es eine Folge  $v_0, e_1, v_1, e_2, \dots, v_{n-1}, e_n, v_n$  ( $n \geq 1$ ) gibt, so daß

$$\bullet \xrightarrow{e_i} \bullet \quad \text{oder} \quad \bullet \xleftarrow{e_i} \bullet \quad \text{für } i = 1, \dots, n$$

$v_{i-1} \quad v_i \qquad v_{i-1} \quad v_i$



**Beispiel 7. (Graphgrammatik)** Sei  $\text{CON} = \langle C, \mathcal{R}, N, S \rangle$  die Graphgrammatik zur Erzeugung aller nichtleeren, zusammenhängenden Graphen mit

- $C_V = C_E = \{\square\}$  und  $N_V = N_E = \emptyset$
- $S = \bullet$
- $\mathcal{R} = \left\{ \begin{array}{l} r_1 : \quad \bullet \supseteq \bullet \subseteq \bullet \rightarrow \bullet \\ r_2 : \quad \bullet \supseteq \bullet \subseteq \bullet \leftarrow \bullet \\ r_3 : \quad \begin{array}{c} \bullet \quad \bullet \\ 1 \quad 2 \end{array} \supseteq \begin{array}{c} \bullet \quad \bullet \\ 1 \quad 2 \end{array} \subseteq \begin{array}{c} \bullet \rightarrow \bullet \\ 1 \quad 2 \end{array} \end{array} \right.$

**Behauptung 1:** Jeder Graph in  $L(\text{CON})$  ist ein nichtleerer, zusammenhängender Graph.

**Beweis:** Zu zeigen ist, daß jeder Graph  $Z$  mit  $S \Rightarrow_{\text{CON}}^* Z$  ein nichtleerer, zusammenhängender Graph ist. Dies wird durch Induktion über die Länge  $n$  der Ableitung gezeigt ( $n$  ist die Anzahl der direkten Ableitungen in  $S \Rightarrow_{\text{CON}}^* Z$ ).

**Induktionsanfang:** Für  $n = 0$  gilt  $Z \cong S = \bullet$ , so daß  $Z$  offensichtlich die geforderte Form hat.

**Induktionsvoraussetzung:** Sei  $n \geq 1$ , und für alle Ableitungen  $S \Rightarrow_{\text{CON}}^* \bar{Z}$  mit weniger als  $n$  Schritten sei  $\bar{Z}$  ein nichtleerer, zusammenhängender Graph.

**Induktionsschritt:** Die gegebene Ableitung hat die Form  $S \Rightarrow_{\text{CON}}^* \bar{Z} \Rightarrow_r Z$  mit  $r \in \mathcal{R}$ , wobei  $S \Rightarrow_{\text{CON}}^* \bar{Z}$   $n - 1$  Schritte enthält.

**Fall 1:**  $r = r_1$ . Dann entsteht  $Z$  aus  $\bar{Z}$  durch Hinzufügen eines Knotens  $v$  und einer Kante  $e$ , so daß  $t_Z(e) = v$  und  $s_Z(e) \in V_{\bar{Z}}$ . Da  $s_Z(e)$  nach Induktionsvoraussetzung mit jedem anderen Knoten in  $\bar{Z}$  verbunden ist, ist  $v$  mit jedem anderen Knoten in  $Z$  verbunden. Also ist  $Z$  zusammenhängend.

**Fall 2:**  $r = r_2$ . Analog zu Fall 1.

**Fall 3:**  $r = r_3$ . Dann entsteht  $Z$  aus  $\bar{Z}$  durch Hinzufügen einer Kante, was den Zusammenhang bewahrt.

**Behauptung 2:** Jeder nichtleere, zusammenhängende Graph ist in  $L(\text{CON})$  enthalten.

**Beweis.** Sei  $Z$  ein nichtleerer, zusammenhängender Graph. Zu zeigen ist, daß es eine Ableitung  $S \Rightarrow_{\text{CON}}^* Z$  gibt. Dies geschieht durch Induktion über  $\text{size}(Z) = |V_Z| + |E_Z|$ , die Anzahl der Knoten und Kanten in  $Z$ .

**Induktionsanfang:** Für  $\text{size}(Z) = 1$  gilt  $Z = \bullet$  und somit  $S \Rightarrow_{\text{CON}}^* Z$  (in 0 Schritten).

**Induktionsvoraussetzung:** Sei  $\text{size}(Z) \geq 2$ , und für jeden nichtleeren, zusammenhängenden Graphen  $\bar{Z}$  mit  $\text{size}(\bar{Z}) < \text{size}(Z)$  gebe es eine Ableitung  $S \Rightarrow_{\text{CON}}^* \bar{Z}$ .

**Induktionsschritt:** Nach dem weiter unten bewiesenen Lemma gibt es in  $Z$  einen Knoten, der genau eine Kante berührt, oder es gibt eine Kante, deren Löschung den Zusammenhang bewahrt.

**Fall 1:** Es gibt einen Knoten  $v$ , der genau eine Kante berührt (d.h.  $t_Z^{-1}(v) = \{e\}$  und  $s_Z^{-1}(v) = \emptyset$  oder  $s_Z^{-1}(v) = \{e\}$  und  $t_Z^{-1}(v) = \emptyset$ ). Sei  $v'$  die Quelle (das Ziel) dieser Kante. Dann gilt  $\bar{Z} \Rightarrow_r Z$  mit  $r = r_1$  oder  $r = r_2$ . Von  $v'$  gibt es eine Verbindung ohne die Kante  $e$  zu jedem anderen Knoten in  $\bar{Z}$  (denn in jeder Verbindung von  $v'$  mit einem Knoten in  $\bar{Z}$  kann die Teilverbindung  $v', e, v, e, v'$  durch  $v'$  ersetzt werden). Somit ist  $\bar{Z}$  zusammenhängend. Wegen  $\text{size}(\bar{Z}) < \text{size}(Z)$  gilt nach Induktionsvoraussetzung  $S \Rightarrow_{\text{CON}}^* \bar{Z} \Rightarrow_{\text{CON}} Z$ .

**Fall 2:** Es gibt eine Kante  $e$  in  $Z$ , deren Löschung den Zusammenhang bewahrt. Sei  $\bar{Z}$  der Teilgraph von  $Z$ , der durch Löschung von  $e$  entsteht. Dann gilt  $\bar{Z} \Rightarrow_{r_3} Z$  und  $\text{size}(\bar{Z}) < \text{size}(Z)$ . Da  $\bar{Z}$  zusammenhängend ist, folgt mit der Induktionsvoraussetzung  $S \Rightarrow_{\text{CON}}^* \bar{Z} \Rightarrow_{\text{CON}} Z$ .  $\square$

**Lemma 8. (zusammenhängende Graphen)** In jedem zusammenhängenden Graphen  $Z$  mit  $\text{size}(Z) \geq 2$  gibt es einen Knoten, der genau eine Kante berührt, oder eine Kante, deren Löschung den Zusammenhang bewahrt.

**Beweis.** Betrachte folgenden Algorithmus:

Eingabe: Ein zusammenhängender Graph  $Z$  mit  $\text{size}(Z) \geq 2$  und  $v \in V_Z$ .

```

begin   Visited :=  $\emptyset$ ;
         while es gibt  $e \in (E_Z - \text{Visited})$  mit  $v = s_Z(e)$  oder  $v = t_Z(e)$ 
           do begin if  $v = s_Z(e)$  then  $v := t_Z(e)$  else  $v := s_Z(e)$ ;
               Visited := Visited  $\cup \{e\}$ 
           end
         end

```

Der Algorithmus terminiert, weil Visited in jedem Schleifendurchlauf vergrößert wird und die Schleife spätestens für Visited =  $E_Z$  verlassen wird.

Nach Abbruch ist  $v$  ein Knoten, der entweder (1) nur eine Kante berührt, oder (2) eine Schleife besitzt, oder (3) mindestens zwei Kanten aus Visited berührt.

Im ersten und zweiten Fall ist die Behauptung offensichtlich erfüllt. Im dritten Fall hat der Algorithmus eine Folge von Kanten besucht, die  $v$  mit sich selbst verbinden, d.h. es gibt eine Verbindung  $v, e_1, v_1, \dots, v_{n-1}, e_n, v_n, e, v$ , wobei  $e$  die vom Algorithmus zuletzt besuchte Kante ist. Somit kann in jeder Verbindung zweier Knoten aus  $Z$ , in der  $e$  vorkommt,  $v, e, v_n$  durch  $v, e_1, v_1, \dots, v_{n-1}, e_n, v_n$  ersetzt werden, bzw.  $v-n, e, v$  durch  $v-n, e_n, v_{n-1}, \dots, v-1, e_1, v$ . Folglich liefert das Löschen von  $e$  einen zusammenhängenden Graphen.  $\square$

## 2.9 Das Bibliothekssystem

**Definition 9. (konsistenter Zustand)** Ein Graph ist ein **konsistenter Zustand der Bibliothek**, wenn die folgenden fünf Bedingungen erfüllt sind:

- (1) Es gibt einen mit BIB markierten Knoten. Dieser ist Quelle von fünf Kanten, deren Ziele mit Autoren, Verlage, Bestellungen, Katalog und Leser markiert sind.
- (2) Die mit Autoren und Verlage markierten Knoten sind Quellen von beliebig vielen Kanten, deren Ziele mit Zeichenketten aus Buchstaben markiert sind. Die mit Bestellungen, Katalog und Leser markierten Knoten sind Quellen von beliebig vielen Kanten, deren Ziele mit Zeichenketten aus Buchstaben und Ziffern markiert sind, so daß das erste Zeichen B, K bzw. L ist. Die aus Katalog herausgehenden Kanten können mit – markiert sein. Die Zielknoten der unter (2) beschriebenen Kanten sind paarweise verschieden.
- (3) Jeder Katalog- und Bestellnummerknoten ist Quelle von drei Kanten, die mit A, T und V markiert sind. Das Ziel der T-Kante ist mit einer Zeichenkette markiert und ist nicht Ziel von anderen Kanten. Die A- und V-Kanten haben Autoren- bzw. Verlagsknoten als Ziele.
- (4) Ist ein Katalognummerknoten Ziel einer mit – markierten Kante, so gibt es genau einen Lesernummerknoten, der mit dem Katalognummerknoten durch zwei entgegengesetzte Kanten verbunden ist.
- (5) Es gibt nur die in (1) bis (4) beschriebenen Knoten und Kanten.

### Bibliotheksoperationen

- (1) Aufnahme eines Autors in das Autorenverzeichnis
- (2) Aufnahme eines Verlages in das Verlagsverzeichnis
- (3) Aufnahme eines Lesers in die Leserkartei
- (4) Bestellung eines Buches beim Verlag
- (5) Katalogisierung eines Buches

- (6) Buchausleihe
- (7) Buchrückgabe
- (8) Abbestellung eines Buches
- (9) Streichung eines Buches aus dem Katalog

## 2.10 Das Graphersetzungssystem BIB

**Beispiel 8. (Graphersetzungssystem)** Sei  $\text{BIB} = \langle C, \mathcal{R} \rangle$  das Graphersetzungssystem mit

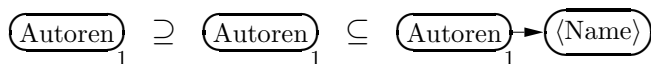
- $C_V = \{\text{BIB}, \text{Autoren}, \text{Verlage}, \text{Bestellungen}, \text{Katalog}, \text{Leser}\} \cup \{a, \dots, z, A, \dots, Z, 0, \dots, 9, \sqcup\}^*$  mit  $\sqcup = \text{Blank}$
- $C_E = \{A, T, V, -, \square\}$  mit  $\square = \text{unmarkiert}$
- $\mathcal{R}$ : siehe unten (Bibliotheksooperationen als unendliche Regelmengen, beschrieben durch neun Regelschemata).

**Frage:**

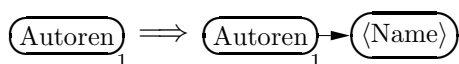
- (1) Lassen sich inkonsistente Zustände aus konsistenten Zuständen ableiten?
- (2) Können alle konsistenten Zustände aus einem geeigneten Anfangszustand abgeleitet werden?

**Regelschemata**

- (1) **AddAutor**( $\langle \text{Name} \rangle$ )



**Kurznotation:**



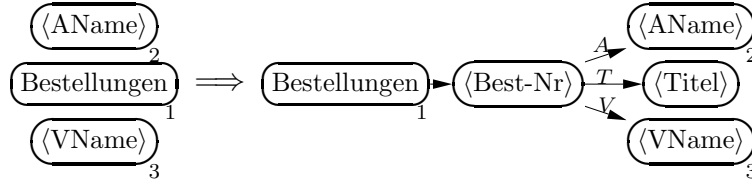
(2) **AddVerlag**( $\langle \text{Name} \rangle$ )



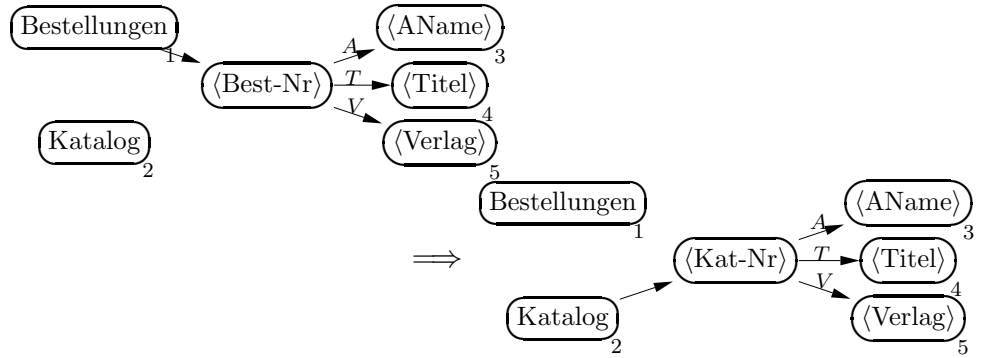
(3) **AddLeser**( $\langle \text{Leser-Nr} \rangle$ )



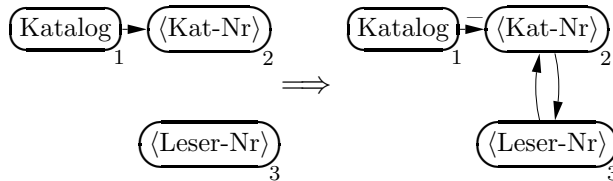
(4) **Bestellen**( $\langle \text{Best-Nr} \rangle, \langle \text{AName} \rangle, \langle \text{Titel} \rangle, \langle \text{VName} \rangle$ )



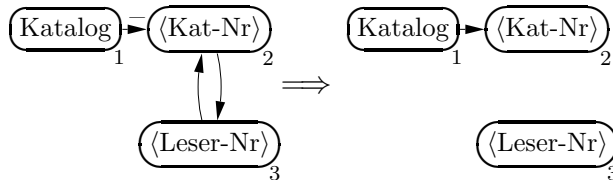
(5) **Katalogisieren**( $\langle \text{Best-Nr} \rangle, \langle \text{Kat-Nr} \rangle$ )

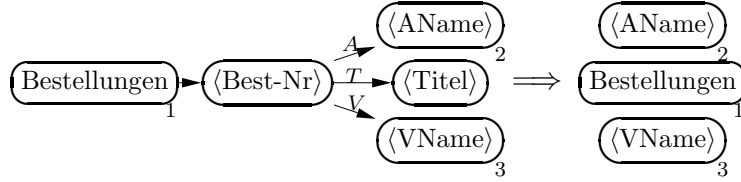
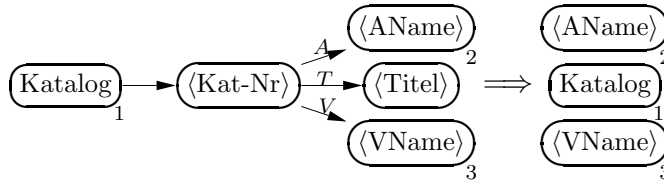


(6) **Ausleihen**( $\langle \text{Kat-Nr} \rangle, \langle \text{Leser-Nr} \rangle$ )



(7) **Abgeben**( $\langle \text{Kat-Nr} \rangle$ )



(8) **Abbestellen**( $\langle \text{Best-Nr} \rangle$ )(9) **Löschen**( $\langle \text{Kat-Nr} \rangle$ )

**Theorem 9. (Korrektheit)** Alle BIB-Regeln überführen konsistente Zustände in konsistente Zustände.

**Beweis.** Sei  $H$  der Graph, der durch Anwendung einer der BIB-Regeln auf einen beliebigen konsistenten Zustand  $G$  entsteht. Es muß gezeigt werden, daß  $H$  ebenfalls ein konsistenter Zustand der Bibliothek ist, d.h. daß  $H$  die Bedingungen (1) bis (5) erfüllt.

- (1) Da keine BIB-Regel den mit BIB markierten Knoten löscht, enthält mit  $G$  auch  $H$  einen solchen Knoten. Da in allen Regeln die mit Autoren, Verlage, Bestellungen, Katalog und Leser markierten Knoten — soweit überhaupt vorhanden — Verklebungsknoten sind, sie also durch die Anwendung der Regeln nicht geändert werden, ist in  $H$  — wie in  $G$  — der BIB-Knoten Quelle von fünf Kanten, deren Ziele mit Autoren, Verlage, Bestellungen, Katalog und Leser markiert sind.
- (2) Die Operationen **AddAutor**, **AddVerlag** und **AddLeser** erweitern die in  $G$  an den Autoren-, Verlage- bzw. Leser-Knoten bereits vorhandenen Kantenbüschel um eine weitere Kante, deren Zielknoten ein korrekt markierter neuer Knoten ist. Andere Teile von  $G$  werden durch die Regeln nicht beeinflusst. In allen anderen Regeln sind die Autoren-, Verlage- bzw. Leser-Knoten Verklebungsknoten, so daß sie durch die Anwendung von Regeln nicht verändert werden.  
Die Operation **Bestellen** erweitert das an dem Bestellungen-Knoten

vorhandene Kantenbüschel um eine Kante, deren Ziel korrekt markiert ist. Dagegen verkleinern die Operationen **Abbestellen** und **Katalogisieren** das an dem Bestellungen-Knoten hängende Kantenbüschel, da die Bestellnummer auf der linken Seite **kein** Verklebungsknoten ist, also beim Anwenden gelöscht wird; es bleibt aber ein erlaubtes Kantenbüschel, da insbesondere die Kantenzahl auch Null sein kann.

Durch die Operation **Katalogisieren** wird das an dem Katalog-Knoten hängende Kantenbüschel erweitert: der Kat-Nr-Knoten auf der rechten Seite der Regel ist nicht Verklebungsknoten; beim Anwenden der Regel wird also ein Kat-Nr-Knoten und eine Kante vom Katalog-Knoten zum Kat-Nr-Knoten eingefügt.

Die Operation **Löschen** wirkt im Katalog-Knoten wie **Abbestellen** auf den Bestellungen-Knoten. Die Operationen **Ausleihen** und **Abgeben** ändern an den von den Katalog- und Bestellungen-Knoten ausgehenden Kantenbüscheln nichts, da der Kat-Nr-Knoten Verklebungsknoten ist, Bestellnummern in den Ansätzen nicht vorkommen und die Markierungsänderung der Kante vom Katalog- zum Kat-Nr-Knoten nach (2) erlaubt ist.

- (3) Keine der Operationen ändert die Kanten, die in  $G$  einen Best-Nr- oder Kat-Nr-Knoten verlassen, oder ihre Ziele, es sei denn, der Quellknoten wird gelöscht. Es muß also nur geprüft werden, ob in  $H$  neu hinzukommende Bestell- oder Katalogknoten Quelle von drei Kanten sind, die mit A, T und V markiert sind. Das betrifft lediglich die Operationen **Bestellen** und **Katalogisieren**,

bei denen auf der rechten Seite, und damit auch in  $H$ , die neue Best-Nr bzw. Kat-Nr Quelle von drei Kanten ist, die mit A, T und V markiert sind. In den Regeln sind der AName- und der VName-Knoten Verklebungsknoten, so daß sie wegen der Konsistenz von  $G$  zum Autoren- bzw. Verlagsverzeichnis gehören.

Der Titel-Knoten kommt beim **Bestellen** neu hinzu, so daß er nur mit dem Best-Nr-Knoten verbunden ist, jedoch nicht Ziel von anderen Kanten ist; er wird beim **Katalogisieren** von der Best-Nr zur Kat-Nr übergeben, so daß auch in diesem Fall genau eine Kante den Titel als Ziel hat.

- (4) Nur die Operationen **Ausleihen** und **Abgeben** verändern die in (4)



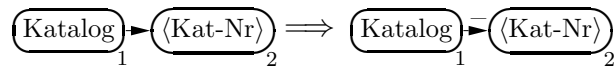
angesprochenen lokalen Eigenschaften, aber genau in der vorgeschriebenen Form: **Ausleihen** ersetzt die Kante durch eine mit  $-$  markierte Kante, verbindet jedoch gleichzeitig den Kat-Nr-Knoten und den Leser-Nr-Knoten durch zwei entgegengesetzte Kanten; **Abgeben** löscht die Verbindungen, ersetzt aber die mit  $-$  markierte Kante durch eine unmarkierte Kante.

(5)  $H$  hat nur die in (1) bis (4) beschriebenen Knoten und Kanten.

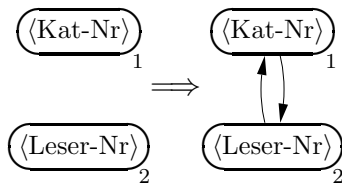
□

**Bemerkung.** Konsistenzerhaltung ist **nicht** selbstverständlich:

(6.1) **Ausleihen1**( $\langle \text{Kat-Nr} \rangle$ )

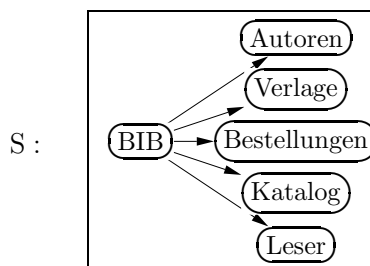


(6.2) **Ausleihen2**( $\langle \text{Kat-Nr} \rangle, \langle \text{Leser-Nr} \rangle$ )



**Ausleihen1** und **Ausleihen2** überführen konsistente Zustände in inkonsistente Zustände.

**Theorem 10. (Vollständigkeit)** Aus dem konsistenten Zustand



lassen sich mit Hilfe der Regeln **AddAutor**, **AddVerlag**, **AddLeser**, **Bestellen**, **Katalogisieren** und **Ausleihen** alle konsistenten Zustände der Bibliothek ableiten.

**Beweis.** (durch vollständige Induktion über die Zahl der Knoten und Kanten  $|G|$  konsistenter Zustände  $G$ ). Sei  $G$  ein beliebiger konsistenter Zustand.

(1)  $|G| = 11$  impliziert  $G \cong S$  und  $S \Rightarrow^0 G$ .

(2)  $|G| > 11$ .

**Fall 1.** Es gibt in  $G$  eine mit – markierte Kante von dem Katalog-Knoten zu einem Kat-Nr-Knoten mit Markierung K1. Wegen der Konsistenz von  $G$  gibt es einen Leser-Nr-Knoten mit Markierung L1 und zwei Kanten von dem K1-Knoten zu dem L1-Knoten und zurück. Damit ist die Regel **Abgeben**(K1) auf  $G$  anwendbar. Nach dem Korrektheitssatz ist der resultierende Graph  $G'$  konsistent.  $G'$  hat zwei Kanten weniger als  $G$ , d.h.  $|G'| < |G|$ . Nach Induktionsvoraussetzung existiert eine Ableitung  $S \Rightarrow^* G'$  mit den genannten Regeln. Auf  $G'$  ist die Regel **Ausleihen**(K1,L1) anwendbar und liefert  $G$ , da **Ausleihen**(K1,L1) und **Abgeben**(K1) invers sind. Damit existiert eine Ableitung  $S \Rightarrow^* G' \Rightarrow G$  mit den genannten Regeln.

**Fall 2.** Es gibt in  $G$  eine unmarkierte Kante von dem Katalog-Knoten zu einem Kat-Nr-Knoten mit Markierung K1. Wegen der Konsistenz von  $G$  gibt es drei Kanten: zu einem AName-Knoten mit Markierung A1, einem Titel-Knoten mit Markierung T1 und zu einem VName-Knoten mit Markierung V1. Damit ist die Regel **Löschen**(K1) auf  $G$  anwendbar. Nach dem Korrektheitssatz ist der resultierende Graph  $G'$  konsistent.  $G'$  hat vier Kanten und einen Knoten weniger als  $G$ . Nach Induktionsvoraussetzung existiert eine Ableitung  $S \Rightarrow^* G'$  mit den genannten Regeln. Die Löschung kann durch eine Bestellung mit beliebiger Bestellnummer B1 und einer Katalogisierung mit Katalognummer K1 aufgehoben werden. Die Anwendung der Regeln **Bestellen**(B1,A1,T1,V1) und **Katalogisieren**(B1,K1) liefert  $G$ . Damit existiert eine Ableitung  $S \Rightarrow^* G' \Rightarrow^2 G$  mit den genannten Regeln.

**Fall 3.**  $G$  enthält einen Best-Nr-Knoten (mit Markierung B1). Dann läßt sich mit **Abbestellen**(B1) analog vorgehen.

**Fall 4.**  $G$  enthält einen Leser-Nr-Knoten (mit Markierung L1). Dann erhält man einen kleineren konsistenten Zustand, wenn man den L1-Knoten mit anhängender Kante löscht. (o.B.d.A. kann man annehmen, daß der L1-Knoten mit keinem Kat-Nr-Knoten verbunden ist. Dieser Fall ist bereits im Fall 1 behandelt worden). Nach Induktionsvoraussetzung existiert eine Ableitung  $S \Rightarrow^* G'$  mit den genannten Regeln. Nun ist auf  $G'$  die Regel **AddLeser**(L1) anwendbar und liefert  $G$ .

**Fall 5.**  $G$  enthält einen AName-Knoten (mit Markierung A1). (analog zu Fall 4)

**Fall 6.**  $G$  enthält einen VName-Knoten (mit Markierung V1). (analog zu Fall 4)

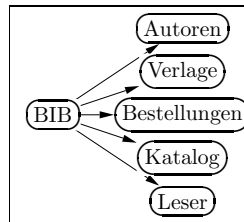
Trifft auf einen konsistenten Zustand  $G$  keiner der sechs Fälle zu, so muß  $G$  mit dem Graphen  $S$  übereinstimmen. Dieser Fall ist in 1.) behandelt.  $\square$

## 2.11 Die Graphgrammatik BIB

**Beispiel 9. (Graphgrammatik)** Sei  $\text{BIB} = \langle C, \mathcal{R}, N, S \rangle$  die Graphgrammatik zur Erzeugung aller konsistenten Zustände mit

- $C_V = \{\text{BIB}, \text{Autoren}, \text{Verlage}, \text{Bestellungen}, \text{Katalog}, \text{Leser}\} \cup \{a, \dots, z, A, \dots, Z, 0, \dots, 9, \sqcup\}^*$  mit  $\sqcup = \text{Blank}$
- $C_E = \{A, T, V, -, \square\}$  mit  $\square = \text{unmarkiert}$
- $\mathcal{R}$ : siehe oben (Bibliotheksooperationen als unendliche Regelmengende, beschrieben durch neun Regelschemata).
- $N_V = N_E = \emptyset$

•  $S :$



### 3 Eigenschaften von Ableitungen

#### Eigenschaften von Ableitungen

Sei  $G_0 \Rightarrow_{r_1, g_1} G_1 \Rightarrow_{r_2, g_2} \dots \Rightarrow_{r_n, g_n} G_n$  eine Ableitung mit Diagrammen

$$\begin{array}{ccccc} L_i & \supseteq & K_i & \subseteq & R_i \\ g_i \downarrow & & d_i \downarrow & & \downarrow h_i \\ G_{i-1} & \supseteq & D_i & \subseteq & G_i \end{array} \quad (i = 1, \dots, n)$$

**Lemma 11. (keine Seiteneffekte)** Sei  $G_0 \Rightarrow_{r_1, g_1} G_1 \Rightarrow_{r_2, g_2} \dots \Rightarrow_{r_n, g_n} G_n$  eine Ableitung. Dann gilt:

- (1) Für alle  $x \in G_0$ :  $x \in G_n$  oder  $x \in g_i(L_i - K_i)$  für ein  $i \in \{1, \dots, n\}$ .
- (2) Für alle  $x \in G_n$ :  $x \in G_0$  oder  $x \in h_i(R_i - K_i)$  für ein  $i \in \{1, \dots, n\}$ .

**Lemma 12. (Umkehrbarkeit)** Sei  $G_0 \Rightarrow_{r_1, g_1} G_1 \Rightarrow_{r_2, g_2} \dots \Rightarrow_{r_n, g_n} G_n$  eine Ableitung. Dann gibt eine Umkehrableitung  $G_n \Rightarrow_{r_n^{-1}, h_n} G_{n-1} \Rightarrow_{r_{n-1}^{-1}, h_{n-1}} \dots \Rightarrow_{r_1^{-1}, h_1} G_0$  mit  $r_i^{-1} = \langle R_i \leftarrow K_i \rightarrow L_i \rangle$  für  $i = 1, \dots, n$ .

Jeder Schritt  $G_i \Rightarrow_{r_i^{-1}, h_i} G_{i-1}$  ist gegeben durch:

$$\begin{array}{ccccc} R_i & \supseteq & K_i & \subseteq & L_i \\ h_i \downarrow & & d_i \downarrow & & \downarrow g_i \\ G_i & \supseteq & D_i & \subseteq & G_{i-1} \end{array}$$

(Löschen und Verkleben sind invers zueinander.)

#### 3.1 Einbettbarkeit

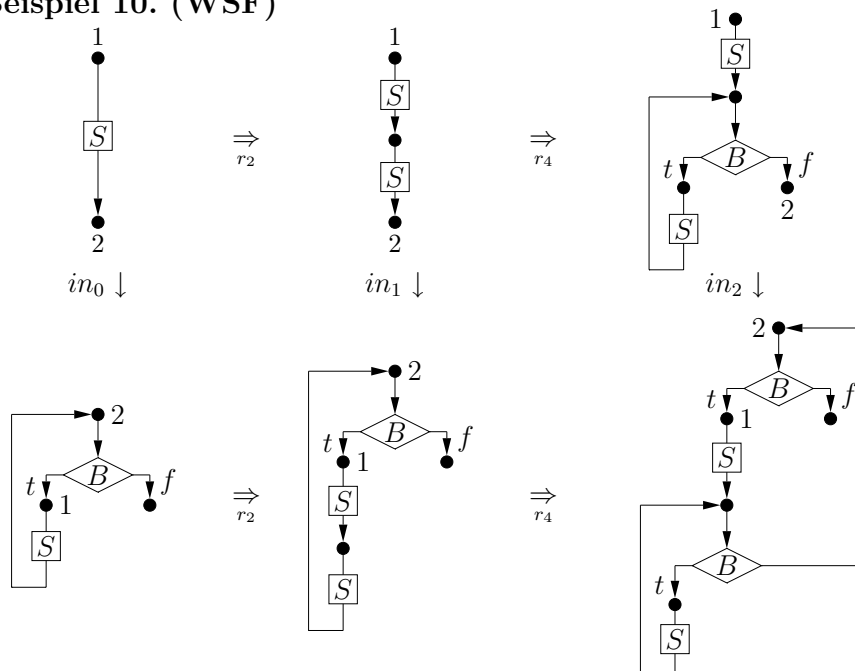
Gegeben:

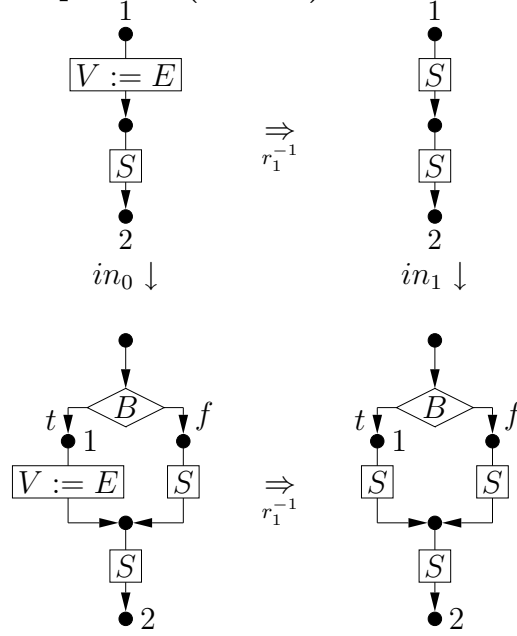
$$\begin{array}{c}
 G_0 \xRightarrow{r_1} G_1 \xRightarrow{r_2} \dots \xRightarrow{r_n} G_n \\
 \downarrow \text{in} \\
 H_0
 \end{array}$$

wobei  $in$  injektiv ist.

**Frage:** Gibt es eine Ableitung  $H_0 \xRightarrow{r_1} H_1 \xRightarrow{r_2} \dots \xRightarrow{r_n} H_n$ ? Wenn ja, lässt sich  $H_n$  direkt aus  $G_n$  konstruieren, d.h. ohne die Ableitungsschritte zu wiederholen?

**Beispiel 10. (WSF)**



**Beispiel 11. (WSF<sup>-1</sup>)**

$r_2^{-1}$  wegen Kontaktbedingung nicht anwendbar!

**Problem:**  $G_1 \Rightarrow G_2$  löscht einen Knoten, der in  $H_1$  Ziel einer Kante aus  $H_1 - in_1(G_1)$  ist.

**Gesucht:** Bedingung, daß Kanten aus  $H_0 - in_0(G_0)$  keine Knoten berühren, die später gelöscht werden.

**Definition 10. (Rand, Persistent)** Sei  $in : G \rightarrow H$  ein injektiver Graphenmorphismus.

$\text{Rand}(in) \subseteq G$  bezeichne den diskreten Graphen mit Knotenmenge  $\{v \in V_G \mid \exists e \in E_H - in_E(E_G) : s_H(e) = in_V(v) \vee t_H(e) = in_V(v)\}$ .

$\text{Kontext}(in) \subseteq H$  bezeichne den Graphen  $H - in(G - \text{Rand}(in))$ .

$\overline{in} : \text{Rand}(in) \rightarrow \text{Kontext}(in)$  sei die Einschränkung von  $in$ .

$$\begin{array}{ccc} G & \supseteq & \text{Rand}(in) \\ in \downarrow & & \downarrow \overline{in} \\ H & \supseteq & \text{Kontext}(in) \end{array}$$

$$\text{Persistent}(G_0 \Rightarrow^* G_n) = \bigcap_{i=1}^n D_i$$

**Theorem 13. (Einbettungssatz)** Sei  $G_0 \Rightarrow_{r_1} G_1 \Rightarrow_{r_2} \dots \Rightarrow_{r_n} G_n$  eine Ableitung und  $in : G_0 \rightarrow H_0$  ein injektiver Graphenmorphismus mit

$$\text{Rand}(in) \subseteq \text{Persistent}(G_0 \Rightarrow^* G_n).$$

Dann gibt es eine Ableitung  $H_0 \Rightarrow_{r_1} H_1 \Rightarrow_{r_2} \dots \Rightarrow_{r_n} H_n$  derart, daß  $H_n$  die Verklebung von  $\text{Kontext}(in)$  und  $G_n$  gemäß  $\overline{in}$  ist:

$$\begin{array}{ccccc} G_0 & \supseteq & \text{Rand}(in) & \subseteq & G_n \\ \downarrow in & & \downarrow \overline{in} & & \downarrow h \\ H_0 & \supseteq & \text{Kontext}(in) & \subseteq & H_n \end{array}$$

**Beweis.** Sei  $G_0 \Rightarrow_{r_1} G_1 \Rightarrow_{r_2} \dots \Rightarrow_{r_n} G_n$  eine Ableitung und  $in : G_0 \rightarrow H_0$  ein injektiver Graphenmorphismus mit  $\text{Rand}(in) \subseteq \text{Persistent}(G_0 \Rightarrow^* G_n)$ . Ziel ist, eine Ableitung  $H_0 \Rightarrow_{r_1} H_1 \Rightarrow_{r_2} \dots \Rightarrow_{r_n} H_n$  zu konstruieren.

**Ausgangssituation:**

**Konstruktion:**

$\text{Rand}(in) \subseteq G_0$ . Unter Ausnutzung der Voraussetzung erhält man für  $i = 1, \dots, n$  auch

$$\text{Rand}(in) \subseteq \text{Persistent}(G_0 \Rightarrow^* G_n) \subseteq \text{Persistent}(G_{i-1} \Rightarrow^* G_n) \subseteq D_i \subseteq G_i.$$

$H_0$  PO-Objekt von  $G_0$  und  $\text{Kontext}(in)$  gemäß  $\overline{in}$ . Sei nun  $H_{i-1}$  PO-Objekt von  $G_{i-1}$  und  $\text{Kontext}(in)$  gemäß  $\overline{in}$ . Konstruiere  $E_i$  als PO-Objekt von  $D_i$  und  $\text{Kontext}(in)$  gemäß  $\overline{in}$ . Dann existiert eindeutig ein Graph-Morphismus  $E_i \rightarrow H_{i-1}$  derart, daß  $D_i \rightarrow E_i \rightarrow H_{i-1} = D_i \rightarrow G_{i-1} \rightarrow H_{i-1}$  und  $\text{Kontext}(in) \rightarrow E_i \rightarrow H_{i-1} = \text{Kontext}(in) \rightarrow H_{i-1}$ . Nach dem Dekompositionslemma für PO's ist  $H_{i-1}$  PO-Objekt von  $G_{i-1}$  und  $E_i$  gemäß  $D_i \rightarrow E_i$ .

Konstruiere nun  $H_i$  PO-Objekt von  $G_i$  und  $E_i$  gemäß  $D_i \rightarrow E_i$ . Dann ist nach dem Kompositionslemma für PO's  $H_i$  PO-Objekt von  $G_i$  und  $\text{Kontext}(in)$  gemäß  $\overline{in}$ .

**Zusammensetzung:**

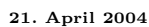
Definiere, für  $i = 1, \dots, n$ ,  $L_i \rightarrow H_{i-1} = L_i \rightarrow G_{i-1} \rightarrow H_{i-1}$ ,  $K_i \rightarrow E_i = K_i \rightarrow D_i \rightarrow E_i$  und  $R_i \rightarrow H_i = R_i \rightarrow G_i \rightarrow H_i$ . Dann folgt mit

☐

Gegeben:

wobei  $in$  injektiv ist.

### Beispiel 12. ( $\text{WSF}^{-1}$ )





**Definition 11. (Used)** Sei  $\text{Used}(H_0 \Rightarrow^* H_n) = \bigcup_{i=1}^n (h_i(L_i) \cap \text{Persistent}(H_0 \Rightarrow^* H_{i-1}))$ , wobei  $\text{Persistent}(H_0 \Rightarrow^* H_0) = H_0$ .

**Theorem 14. (Einschränkungssatz)** Sei  $H_0 \Rightarrow_{r_1} H_1 \Rightarrow_{r_2} \dots \Rightarrow_{r_n} H_n$  eine Ableitung und  $in : G_0 \rightarrow H_0$  ein injektiver Graphenmorphismus mit

$$\text{Used}(H_0 \Rightarrow^* H_n) \subseteq in(G_0).$$

Dann gibt es eine Ableitung  $G_0 \Rightarrow_{r_1} G_1 \Rightarrow_{r_2} \dots \Rightarrow_{r_n} G_n$  mit  $G_n = H_n - (\text{Kontext}(in) - in(\text{Rand}(in)))$ . Dabei sind  $\text{Rand}(in)$  und  $\text{Kontext}(in)$  analog zum Einbettungssatz definiert.

$$\begin{array}{ccccc} G_0 & \supseteq & \text{Rand}(in) & \subseteq & G_n \\ \downarrow in & & \downarrow \overline{in} & & \downarrow h \\ H_0 & \supseteq & \text{Kontext}(in) & \subseteq & H_n \end{array}$$

**Beweis.** Sei  $H_0 \Rightarrow_{r_1} H_1 \Rightarrow_{r_2} \dots \Rightarrow_{r_n} H_n$  eine Ableitung und  $in : G_0 \rightarrow H_0$  ein injektiver Graphenmorphismus mit  $\text{Used}(H_0 \Rightarrow^* H_n) \subseteq in(G_0)$ . Ziel ist es, eine Ableitung  $G_0 \Rightarrow_{r_1} G_1 \Rightarrow_{r_2} \dots \Rightarrow_{r_n} G_n$  zu konstruieren.

**Ausgangssituation:**

**Konstruktion:**

Nach Konstruktion von  $\text{Kontext}(in)$  ist  $H_0$  Verklebung von  $G_0$  und  $\text{Kontext}(in)$  gemäß  $\text{Rand}(in) \rightarrow \text{Kontext}(in)$ .

Nach Definition von  $\text{Used}$  und der Voraussetzung gilt  $h_1(L_1) \subseteq \text{Used}(H_0 \Rightarrow^* H_n) \subseteq in(G_0)$ . Definiere nun  $g_1 : L_1 \rightarrow G_0$  für alle Punkte  $x \in L_1$  durch  $g_1(x) = y$  falls  $y \in \epsilon^{-1} h_1(x)$  ist. Da  $h_1(L_1) \subseteq in(G_0)$  und  $in$  injektiv ist, ist  $g_1$  wohldefiniert. Dann ist  $L_1 \rightarrow G_0$  ein Graph-Morphismus mit  $L_1 \rightarrow G_0 \rightarrow H_0 = L_1 \rightarrow H_0$ . Wegen ... gilt  $\text{Kontext}(in) \subseteq E_1 \subseteq H_0$ .

Sei nun  $H_{i-1}$  Verklebung von  $G_{i-1}$  und  $\text{Kontext}(in)$  gemäß  $\text{Rand}(in) \rightarrow \text{Kontext}(in)$  und  $L_i \rightarrow G_{i-1}$  ein Graph-Morphismus mit  $L_i \rightarrow G_{i-1} \rightarrow H_{i-1} = L_i \rightarrow H_{i-1}$  und  $\text{Kontext}(in) \subseteq E_i \subseteq H_{i-1}$ . Konstruiere nun  $D_i$  als PB-Objekt von  $G_{i-1} \rightarrow H_{i-1} \leftarrow E_i$ . Dann existieren eindeutig Graph-Morphismen  $K_i \rightarrow D_i$  und  $\text{Rand}(in) \rightarrow D_i$  derart, daß  $K_i \rightarrow D_i \rightarrow E_i =$

$K_i \rightarrow E_i$ ,  $K_i \rightarrow D_i \rightarrow G_{i-1} = K_i \rightarrow L_i \rightarrow G_{i-1}$ ,  $\text{Rand}(in) \rightarrow D_i \rightarrow E_i = \text{Rand}(in) \rightarrow \text{Kontext}(in) \rightarrow E_i$  und  $\text{Rand}(in) \rightarrow D_i \rightarrow G_{i-1} = \text{Rand}(in) \rightarrow G_{i-1}$ .

Da nach Voraussetzung  $K_i L_i E_i H_{i-1}$  ein PO ist und  $L_1 \rightarrow H_{i-1} = L_1 \rightarrow G_{i-1} \rightarrow H_{i-1}$  ist, sind die Graph-Morphismen  $G_{i-1} \rightarrow H_{i-1}$  und  $E_i \rightarrow H_{i-1}$  gemeinsam surjektiv. Darüberhinaus sind  $G_{i-1} \rightarrow H_{i-1}$  und  $E_i \rightarrow H_{i-1}$  injektiv. Da weiterhin das Diagramm  $D_i E_i G_{i-1} H_{i-1}$  ein PB-Diagramm ist, ist es nach dem Pullback-Pushout-Lemma auch ein PO-Diagramm. Da die Diagramme  $K_i L_i E_i H_{i-1}$ ,  $\text{Rand}(in) \text{Kontext}(in) G_{i-1} H_{i-1}$  und  $D_i E_i G_{i-1} H_{i-1}$  PO's sind,  $K_i L_i D_i G_{i-1}$  und  $\text{Rand}(in) \text{Kontext}(in) D_i E_i$  kommutativ sind, und  $D_i \rightarrow E_i$  und  $D_i \rightarrow G_{i-1}$  injektiv sind, folgt mit dem speziellen Dekompositionslemma für PO's, daß die Diagramme  $K_i L_i D_i G_{i-1}$  und  $\text{Rand}(in) \text{Kontext}(in) D_i E_i$  auch PO's sind.

Konstruiere nun  $G_i$  als PO-Objekt von  $R_i$  und  $D_i$  gemäß  $K_i \rightarrow D_i$ . Dann existieren eindeutig ein Graph-Morphismus  $G_i \rightarrow H_i$  derart, daß  $R_i \rightarrow G_i \rightarrow H_i = R_i \rightarrow H_i$  und  $D_i \rightarrow G_i \rightarrow H_i = D_i \rightarrow E_i \rightarrow H_i$ . Da  $K_i R_i E_i H_i$  und  $K_i R_i D_i G_i$  PO's sind und  $D_i G_i E_i H_i$  kommutativ ist, ist nach dem Dekompositionslemma für PO's  $D_i G_i E_i H_i$  ein PO. Da  $\text{Rand}(in) \text{Kontext}(in) D_i E_i$  und  $D_i E_i G_i H_i$  PO's sind, ist nach dem Kompositionslemma für PO' auch  $\text{Rand}(in) \text{Kontext}(in) G_i H_i$  ein PO.

$$\begin{array}{ccccc}
 & & K_i & \longrightarrow & R_i \\
 & & \downarrow & & \downarrow \\
 \text{Rand}(in) & \longrightarrow & D_i & \longrightarrow & G_i \\
 \downarrow & & \downarrow & & \downarrow \\
 \text{Kontext}(in) & \longrightarrow & E_i & \longrightarrow & H_i
 \end{array}$$

Folglich ist  $H_i$  Verklebung von  $G_i$  und  $\text{Kontext}(in)$  gemäß  $\text{Rand}(in) \rightarrow \text{Kontext}(in)$ . ... und  $L_{i+1} \rightarrow G_i$  ein Graph-Morphismus mit  $L_{i+1} \rightarrow G_i \rightarrow H_i = L_{i+1} \rightarrow H_i$  und  $\text{Kontext}(in) \subseteq E_{i+1} \subseteq H_i$ .

**Ergebnis:**

Folglich gibt es eine Ableitung  $G_0 \Rightarrow_{r_1} G_1 \Rightarrow_{r_2} \dots \Rightarrow_{r_n} G_n$  mit  $G_n = H_n - (\text{Kontext}(in) - in(\text{Rand}(in)))$ .  $\square$

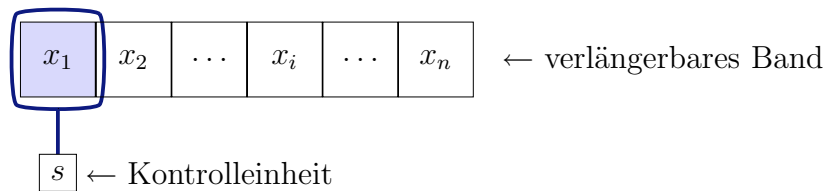
$$\begin{array}{ccccccc}
 & L_1 & \longleftarrow K_1 & \longrightarrow R_1 & L_2 & \longleftarrow K_2 & \longrightarrow R_2 \cdots \\
 G_0 & \swarrow & \downarrow (L1) & \downarrow (R1) & \swarrow & \downarrow (L2) & \downarrow (R2) \\
 H_0 & \longleftarrow & E_1 & \longrightarrow H_1 & \longleftarrow & E_2 & \longrightarrow H_2 \cdots
 \end{array}$$

## 4 Mächtigkeit von Graphersetzungs-systemen

**Frage:** Was können Graphersetzungs-systeme? Wie mächtig sind sie? Welche Probleme lassen sich mit ihnen lösen?

**Antwort:** Alles Berechenbare, denn Graphersetzungs-systeme können Turingmaschinen simulieren!

### Turingmaschinen



$S$  endliche Zustandsmenge

$X$  endliches Bandalphabet mit  $B \in X$

$P \subseteq S \times X \times S \times X \times \{l, r, n\}$

**Anfangskonfiguration:** Kontrolleinheit ganz links, Anfangszustand  $s_0$ .

**Arbeitsweise:** Für  $(s, x, s', x', \beta) \in P$ , Zustand  $s$  und gelesenes Zeichen  $x$  wird in den Zustand  $s'$  übergegangen,  $x$  durch  $x'$  überschrieben und die Kontrolleinheit gemäß  $\beta$  bewegt.

**Konfigurationsbeschreibung:**  $usv \in X^* \cdot S \cdot X^+$

**Überführungsrelation:**

$usv \xrightarrow{P} u's'v'$  durch  $(s, x, s', x', \beta) \in P$  falls einer der folgenden Fälle gilt:

$$\beta = n, u' = u, v = x\bar{v}, v' = x'\bar{v}$$

$$\beta = l, u' = \bar{u}y, u' = \bar{u}, v = x\bar{v}, v' = yx'\bar{v}$$

$$\beta = l, u = \lambda, u' = \lambda, v = x\bar{v}, v' = Bx'\bar{v}$$

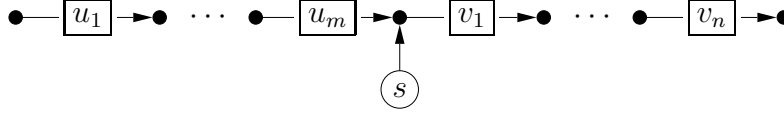
$$\beta = r, v = xy\bar{v}, u' = ux', v' = y\bar{v}$$

$$\beta = r, v = x, u' = ux', v' = B$$

Hierbei ist  $y \in X$ ,  $\bar{u}, \bar{v} \in X^*$ ;  $\lambda$  bezeichnet das leere Wort.

### 4.1 Turingmaschine $TM$ als Graphersetzungssystem $\mathcal{G}(TM)$

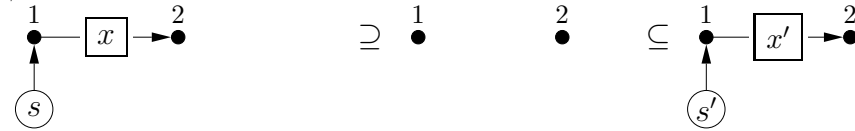
Konfigurationsbeschreibung  $usv$  als Graph  $G(usv)$ :



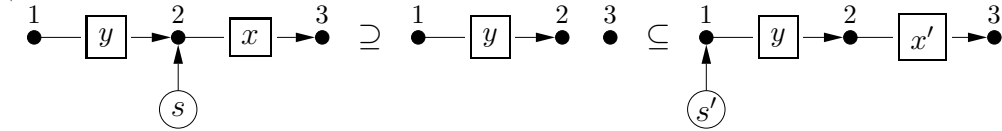
Programm  $P$  als Regelmenge  $\mathcal{R}(P)$ :

Übersetzung von  $(s, x, s', x', \beta) \in P$ :

$\beta = n$ :

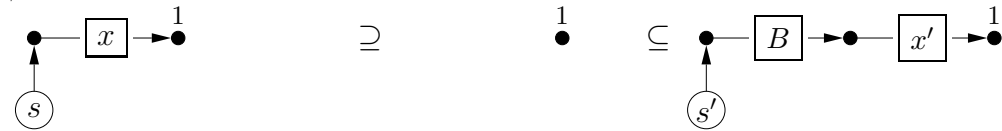


$\beta = l$ :

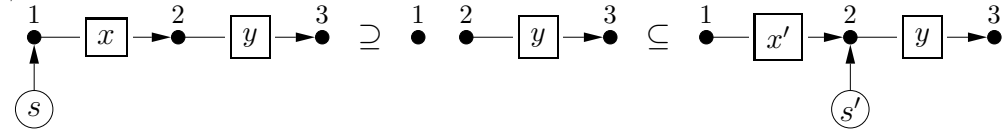


für alle  $y \in X$

$\beta = l$ :



$\beta = r$ :



für alle  $y \in X$

$\beta = r$ :



**Theorem 15.** (Simulation von Turingmaschinen) Für alle Konfigurationen

$usv \in X^*SX^+$  und alle Graphen  $H'$  gilt:  $G(usv) \Rightarrow_{\mathcal{R}(P)} H'$  genau dann wenn  $H' = G(u's'v')$  für eine Nachfolgekongfiguration  $u's'v'$  von  $usv$  ist.

**Beweis.** als Übung □

**Folgerung.**

1. Jede berechenbare Funktion läßt sich durch ein Graphersetzungssystem berechnen.
2. Jede Typ-0-Sprache läßt sich von einem Graphersetzungssystem erkennen.

**Beweis.** Die Aussagen folgen unmittelbar aus den entsprechenden für Turingmaschinen □

## 4.2 Übersetzung von Chomsky-Grammatiken

**Definition 12. (Chomsky-Grammatik)** Eine **Chomsky-Grammatik (Typ-0-Grammatik)**  $G = (N, T, P, S)$  besteht aus endlichen Mengen  $N$ ,  $T$  von nichtterminalen bzw. terminalen Symbolen, einer endlichen Menge  $P \subseteq (N \cup T)^+ \times (N \cup T)^*$  von Produktionen und einem Startsymbol  $S \in N$ . Anwendung von  $(u, v) \in P$ :  $xuy \rightarrow_P xvy$  für alle  $x, y \in (N \cup T)^*$  **erzeugte Sprache**:  $L(G) = \{w \in T^* \mid S \rightarrow_P^* w\}$

**Repräsentation von Wörtern durch Graphen:**

$$\begin{aligned} w = x_1 \dots x_n &\quad \mapsto \quad w^\bullet = \bullet \text{---} \boxed{x_1} \text{---} \bullet \text{---} \boxed{x_2} \text{---} \bullet \dots \bullet \text{---} \boxed{x_n} \text{---} \bullet \\ w = \lambda &\quad \mapsto \quad w^\bullet = \bullet \end{aligned}$$

**Produktionsmenge  $P$  als Regelmenge  $\mathcal{R}(P)$ :**

für  $(u, v) \in P$  mit  $v \neq \lambda$ :

$$\begin{array}{c} 1 \\ \bullet \text{---} \boxed{u_1} \text{---} \bullet \dots \bullet \text{---} \boxed{u_n} \text{---} \bullet \end{array} \supseteq \begin{array}{c} 1 \quad 2 \\ \bullet \quad \bullet \end{array} \subseteq \begin{array}{c} 1 \quad 2 \\ \bullet \text{---} \boxed{v_1} \text{---} \bullet \dots \bullet \text{---} \boxed{v_n} \text{---} \bullet \end{array}$$

für  $(u, \lambda) \in P$ :

$$\begin{array}{c} 1 \\ \bullet \text{---} \boxed{u_1} \text{---} \bullet \dots \bullet \text{---} \boxed{u_n} \text{---} \bullet \end{array} \supseteq \begin{array}{c} 1 \quad 2 \\ \bullet \quad \bullet \end{array} \subseteq \begin{array}{c} 1 \quad 2 \\ \bullet \text{---} \boxed{=} \text{---} \bullet \end{array}$$

## 4 MÄCHTIGKEIT VON GRAPHERSETZUNGSSYSTEMEN

47

---

für alle  $x \in N \cup T$ :

$$\begin{array}{ccc}
 \overset{1}{\bullet} \rightarrow \boxed{x} \rightarrow \bullet \rightarrow \boxed{=} \rightarrow \overset{2}{\bullet} & \supseteq & \overset{1}{\bullet} \quad \overset{2}{\bullet} \subseteq \overset{1}{\bullet} \rightarrow \boxed{=} \rightarrow \bullet \rightarrow \boxed{x} \rightarrow \overset{2}{\bullet} \\
 \bullet \rightarrow \boxed{=} \rightarrow \overset{1}{\bullet} & \supseteq & \overset{1}{\bullet} \quad \subseteq \quad \overset{1}{\bullet}
 \end{array}$$

Betrachte die Graphgrammatik:  $\mathcal{G}(G) = (C, \mathcal{R}(P), N', S^\bullet)$  mit  $C_V = \{\square\}$  und  $C_E = N \cup T \cup \{=\}$ ,  $N'_V = \emptyset$  und  $N'_E = N \cup \{=\}$  und  $\mathcal{R}(P)$  oben angegeben.

**Theorem 16. (Simulation von Chomsky-0-Grammatiken)** Sei  $G$  eine Chomsky-0-Grammatik und  $\mathcal{G}(G)$  die zugeordnete Graphgrammatik. Dann gilt:

1. Für alle Graphen  $G$  über  $C$  gilt:  $S^\bullet \Rightarrow_{\mathcal{R}(P)} G$  genau dann, wenn  $G = w^\bullet$  für ein  $w \in L(G)$ .
2.  $L(\mathcal{G}(G)) = \{w^\bullet | w \in L(G)\}$ .

**Beweis.** als Übung

□

**Folgerung. (Unentscheidbarkeit)** Die folgenden Probleme sind unentscheidbar:

1. Ist  $L(\mathcal{G})$  leer?
2. Ist  $L(\mathcal{G})$  endlich?
3. Ist  $H$  in  $L(\mathcal{G})$ ?

für beliebige Graphgrammatiken  $\mathcal{G}$  und beliebige Graphen  $H$ .

**Beweis.** Die Aussagen folgen unmittelbar aus dem entsprechenden Unentscheidbarkeitsaussagen für Chomsky-0-Grammatiken. □

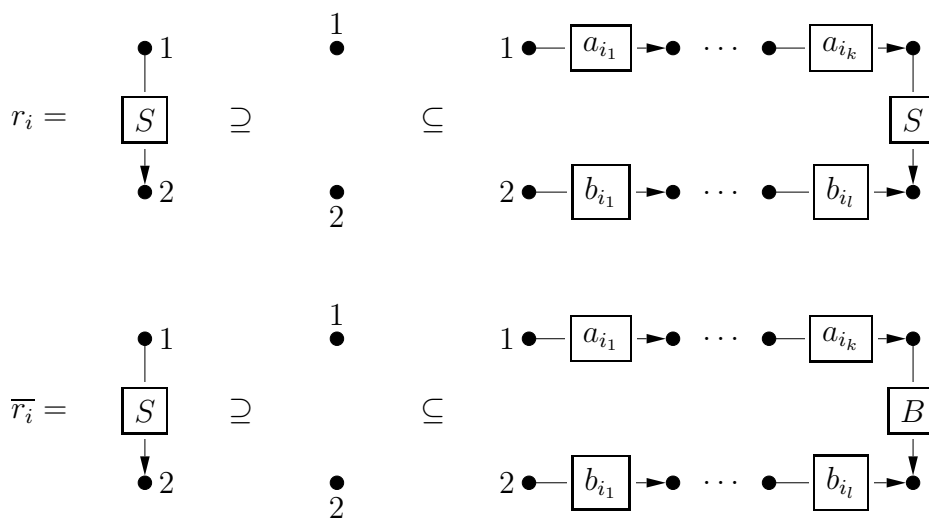
### 4.3 Übersetzung von Instanzen des PCP

**Definition 13. (Postsches Korrespondenzproblem)** Eine **Instanz des Postschen Korrespondenzproblems**  $PCP = (U, V)$  besteht aus zwei Sequenzen  $U = (u_1, \dots, u_n)$  und  $V = (v_1, \dots, v_n)$  von Wörtern über einem Alphabet  $\Sigma$ .

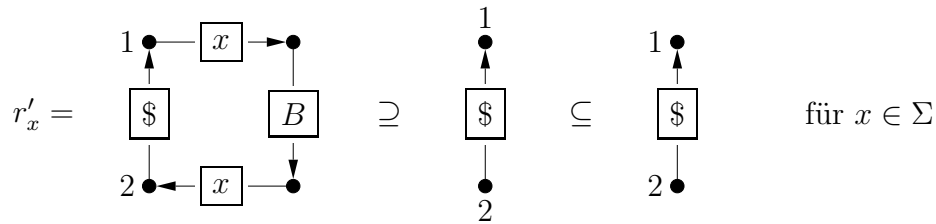
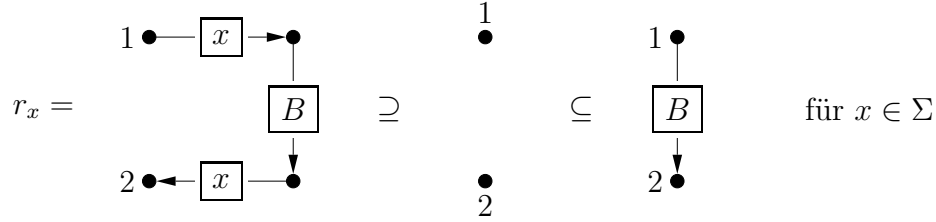
$PCP = (U, V)$  **hat eine Lösung**, wenn es eine Indexfolge  $i_1, \dots, i_m$  mit  $m \geq 1$  gibt, so daß  $u_{i_1} \dots u_{i_m} = v_{i_1} \dots v_{i_m}$ . Die Folge  $i_1, \dots, i_m$  ist eine **Lösung** für die Instanz  $PCP$ .

**Graphersetzungsregeln  $\mathcal{R}(PCP)$**

für eine Instanz  $PCP$  des Postschen Korrespondenzproblems (für  $i = 1, \dots, m$ )

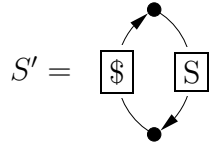






Sei  $\mathcal{G}(PCP) = (C, \mathcal{R}(PCP), N, S^\bullet)$  mit  $C_V = \{\square\}$  und  $C_E = \{S, B\} \cup \Sigma$ ,  $N_V = \emptyset$  und  $N_E = \{S, B\}$  und  $\mathcal{R}(PCP) = \bigcup_{i=1}^m \{r_i, \bar{r}_i\} \cup \{r_x | x \in \Sigma\}$ .

Sei  $\mathcal{G}'(PCP) = (C', \mathcal{R}'(PCP), N, S')$  mit  $C'_V = \{\square\}$  und  $C'_E = \{S, B, \$\} \cup \Sigma$ ,  $\mathcal{R}'(PCP) = \bigcup_{i=1}^m \{r_i, \bar{r}_i\} \cup \{r'_x | x \in \Sigma\}$  und



**Lemma 17.**

1.  $PCP$  hat eine Lösung  
 $\iff L(\mathcal{G}(PCP))$  enthält einen diskreten Graphen.
2.  $PCP$  hat eine Lösung  
 $\iff L(\mathcal{G}'(PCP))$  enthält einen azyklischen Graphen.

**Beweis.** als Übung □

**Folgerung. (Unentscheidbarkeit)** Die folgenden Probleme sind unentscheidbar:

1. Existiert ein Graph  $G$  in  $L(\mathcal{G})$ :  $G$  ist diskret?

2. Existiert ein Graph  $G$  in  $L(\mathcal{G})$ :  $G$  ist azyklisch?

für beliebige Graphgrammatiken  $\mathcal{G}$ .

**Beweis.** Die Unentscheidbarkeitsergebnisse folgen unmittelbar aus der Unentscheidbarkeit des Postschen Korrespondenzproblems.  $\square$

## Literatur

- [Kre93] Hans-Jörg Kreowski. Translations into the graph grammar machine. In Ronan Sleep, Rinus Plasmeijer, Marko van Eekelen, eds., Term Graph Rewriting: Theory and Practice, 171–183. John Wiley, New York, 1993.

## 5 Unabhängigkeit von Ableitungen

### Sequentielle Unabhängigkeit

Gegeben: Zwei direkte Ableitungen der Form  $G \Rightarrow_{r_1} H \Rightarrow_{r_2} M$ .

Frage: Lassen sich die beiden Schritte vertauschen, d.h. gibt es direkte Ableitungen der Form  $G \Rightarrow_{r_2} \bar{H} \Rightarrow_{r_1} M$ ?

**Definition 14. (Sequentielle Unabhängigkeit)** Zwei direkte Ableitungen  $G \Rightarrow_{r_1, g_1} H \Rightarrow_{r_2, g_2} M$  heißen **sequentuell unabhängig**, wenn  $h_1(R_1) \cap g_2(L_2) \subseteq h_1(K_1) \cap g_2(K_2)$  gilt.

$$\begin{array}{ccccccc}
 L_1 & \leftarrow & K_1 & \rightarrow & R_1 & & L_2 & \leftarrow & K_2 & \rightarrow & R_2 \\
 \downarrow g_1 & & \downarrow d_1 & & \searrow h_1 & & \swarrow g_2 & & \downarrow d_2 & & \downarrow h_2 \\
 G & \leftarrow & D_1 & \rightarrow & H & \leftarrow & D_2 & \rightarrow & M
 \end{array}$$

### Lemma 18. (Charakterisierung der sequentiellen Unabhängigkeit)

Zwei direkte Ableitungen  $G \Rightarrow_{r_1, g_1} H \Rightarrow_{r_2, g_2} M$  sind genau dann sequentiell unabhängig, wenn es Graph-Morphismen  $R_1 \rightarrow D_2$  und  $L_2 \rightarrow D_1$  gibt derart, daß  $R_1 \rightarrow D_2 \rightarrow H = R_1 \rightarrow H$  und  $L_2 \rightarrow D_1 \rightarrow H = L_2 \rightarrow H$  gilt.

$$\begin{array}{ccccccc}
 L_1 & \leftarrow & K_1 & \rightarrow & R_1 & & L_2 & \leftarrow & K_2 & \rightarrow & R_2 \\
 \downarrow & & \downarrow & & \searrow & & \swarrow & & \downarrow & & \downarrow \\
 G & \leftarrow & D_1 & \rightarrow & H & \leftarrow & D_2 & \rightarrow & M
 \end{array}$$

(Dashed arrows from  $R_1$  to  $D_2$  and  $L_2$  to  $D_1$  are labeled with  $=$ .)

**Beweis.** “ $\Rightarrow$ ” Sei  $G \Rightarrow_{r_1, g_1} H \Rightarrow_{r_2, g_2} M$  sequentiell unabhängig.

**Beh. 1:**  $h_1(R_1) \subseteq D_2$  und  $g_2(L_2) \subseteq D_1$  mit  $D_1 = H - h_1(R_1 - K_1) = (H - h_1(R_1)) + h_1(K_1)$  und  $D_2 = H - g_2(L_2 - K_2) = (H - g_2(L_2)) + g_2(K_2)$ .

**Bew. 1:** Sei  $x \in h_1(R_1) \subseteq H$ . Im Fall  $x \in H - g_2(L_2)$  ist  $x \in D_2$ . Anderenfalls ist  $x \in g_2(L_2)$  und damit  $x \in h_1(R_1) \cap g_2(L_2) \subseteq h_1(K_1) \cap g_2(K_2) \subseteq D_2$ . Also gilt  $h_1(R_1) \subseteq D_2$ . Analog folgt  $g_2(L_2) \subseteq D_1$ .

**Beh. 2:**  $\bar{h}_1 : R_1 \rightarrow D_2$  mit  $\bar{h}_1(x) = h_1(x)$  für  $x \in R_1$  und  $\bar{g}_2 : L_2 \rightarrow D_1$  mit  $\bar{g}_2(x) = g_2(x)$  für  $x \in L_2$  sind Graph-Morphismen.

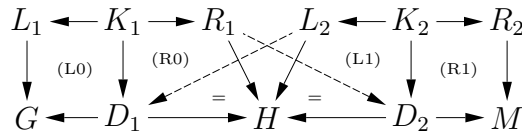
**Bew. 2:** gilt, da  $h_1$  und  $g_2$  Graph-Morphismen sind.

“ $\Leftarrow$ ” Es existieren Graph-Morphismen  $\bar{h}_1 : R_1 \rightarrow D_2$  und  $\bar{g}_2 : L_2 \rightarrow D_1$  mit  $R_1 \rightarrow D_2 \rightarrow H = R_1 \rightarrow H$  und  $L_2 \rightarrow D_1 \rightarrow H = L_2 \rightarrow H$ . Dann ist  $h_1(R_1) \subseteq D_2$  und  $g_2(L_2) \subseteq D_1$ . Folglich ist  $h_1(R_1) \cap g_2(L_2) \subseteq (h_1(R_1) \cap D_1) \cap (g_2(L_2) \cap D_2) \subseteq h_1(K_1) \cap g_2(K_2)$ . Also ist  $G \Rightarrow_{r_1, g_1} H \Rightarrow_{r_2, g_2} M$  sequentiell unabhängig.  $\square$

**Theorem 19. (Sequentielle Unabhängigkeit)** Seien  $G \Rightarrow_{r_1} H$  und  $H \Rightarrow_{r_2} M$  zwei sequentiell unabhängige direkte Ableitungen. Dann gibt es einen Graphen  $\bar{H}$  und sequentiell unabhängige direkte Ableitungen  $G \Rightarrow_{r_1} \bar{H} \Rightarrow_{r_2} M$ .

**Beweis.** Seien  $G \Rightarrow_{r_1} H$  und  $H \Rightarrow_{r_2} M$  sequentiell unabhängig. Dann gibt es Graph-Morphismen  $R_1 \rightarrow D_2$  und  $L_2 \rightarrow D_1$  derart, daß  $R_1 \rightarrow D_2 \rightarrow H = R_1 \rightarrow H$  und  $L_2 \rightarrow D_1 \rightarrow H = L_2 \rightarrow H$  gilt.

**Ausgangssituation:**



**Zerlegung:** Konstruiere nun  $D_0$  als PB-Objekt von  $D_1 \rightarrow H \leftarrow D_2$ . Dann existiert eindeutig ein Graph-Morphismus  $K_1 \rightarrow D_0$  derart, daß  $K_1 \rightarrow D_0 \rightarrow D_1 = K_1 \rightarrow D_1$  und  $K_1 \rightarrow D_0 \rightarrow D_2 = K_1 \rightarrow R_1 \rightarrow D_2$  gilt. Entsprechend existiert eindeutig ein Graph-Morphismus  $K_2 \rightarrow D_0$  derart, daß  $K_2 \rightarrow D_0 \rightarrow D_2 = K_2 \rightarrow D_2$  und  $K_2 \rightarrow D_0 \rightarrow D_1 = K_2 \rightarrow L_2 \rightarrow D_1$  gilt. Konstruiere weiterhin  $\bar{D}_2$  und  $\bar{D}_1$  als PO-Objekte von  $L_1 \leftarrow K_1 \rightarrow D_0$  bzw.  $R_2 \leftarrow K_2 \rightarrow D_0$ . Dann existieren eindeutig Graph-Morphismen  $\bar{D}_2 \rightarrow G$ ,  $\bar{D}_1 \rightarrow M$  derart, daß  $L_1 \rightarrow \bar{D}_2 \rightarrow G = L_1 \rightarrow G$  und  $D_0 \rightarrow \bar{D}_2 \rightarrow G = D_0 \rightarrow D_1 \rightarrow G$  bzw.  $R_2 \rightarrow \bar{D}_1 \rightarrow M = R_2 \rightarrow M$  und  $D_0 \rightarrow \bar{D}_1 \rightarrow M = D_0 \rightarrow D_2 \rightarrow M$  gilt.

Nach Konstruktion ist das Diagramm (R9)=(L9) ein PB; die Diagramme (L2) und (R3) sind PO's. Im folgenden wollen wir zeigen, daß die Diagramme (R9)=(L9), (R2), (R8), (L3) und (L8) PO's sind. Da die Ausgangsdiagramme (R0) und (L1) PO's und  $K_1 \rightarrow R_1$  und  $K_2 \rightarrow L_2$  injektiv sind, sind  $(D_1 \rightarrow H, R_1 \rightarrow H)$  und  $(D_2 \rightarrow H, L_2 \rightarrow H)$  gemeinsam surjektiv und  $D_1 \rightarrow H$  und  $D_2 \rightarrow H$  injektiv (PO-Charakterisierung und Injektivitätslemma). Folglich sind  $(D_1 \rightarrow H, D_2 \rightarrow H)$  gemeinsam surjektiv. Da (R9)=(L9)

$$\begin{array}{ccccc}
L_1 & \leftarrow K_1 & \rightarrow R_1 & & L_2 \leftarrow K_2 \rightarrow R_2 \\
\downarrow & (L2) & \downarrow & (R2) & \downarrow & (L3) & \downarrow & (R3) \\
\overline{D}_2 & \leftarrow D_0 & \rightarrow D_2 & & \overline{D}_1 & \leftarrow D_0 & \rightarrow \overline{D}_1 \\
\downarrow & (L8) & \downarrow & (R9) & \downarrow & (L9) & \downarrow & (R8) \\
G & \leftarrow D_1 & \rightarrow H & & \leftarrow D_2 & \rightarrow M
\end{array}$$

PB ist, ist (R9)=(L9) auch PO (PB-PO-Lemma). Da (R0)=(R2)+(R9) und (R9) PO's sind und  $D_2 \rightarrow H$  injektiv ist, ist (R2) PO (spezielles Dekompositionslemma für PO's). Da (L0)=(L2)+(L8) und (L2) PO's sind, ist (L8) PO (Dekompositionslemma für PO's). Analog folgt, daß die Diagramme (L3) und (R8) PO's sind.

**Zusammensetzung:** Konstruiere nun  $\overline{H}$  als PO-Objekt zu  $\overline{D}_1 \leftarrow D_0 \rightarrow \overline{D}_2$ .

$$\begin{array}{ccccc}
L_2 & \leftarrow K_2 & \rightarrow R_2 & & L_1 \leftarrow K_1 \rightarrow R_1 \\
\downarrow & (L3) & \downarrow & (R3) & \downarrow & (L2) & \downarrow & (R2) \\
D_1 & \leftarrow D_0 & \rightarrow \overline{D}_1 & & \overline{D}_2 & \leftarrow D_0 & \rightarrow D_2 \\
\downarrow & (L8) & \downarrow & (R9) & \downarrow & (L9) & \downarrow & (R8) \\
G & \leftarrow \overline{D}_2 & \rightarrow \overline{H} & & \leftarrow \overline{D}_1 & \rightarrow M
\end{array}$$

Definiere  $L_2 \rightarrow G = L_2 \rightarrow D_1 \rightarrow G$ ,  $K_2 \rightarrow \overline{D}_2 = K_2 \rightarrow D_0 \rightarrow \overline{D}_2$ ,  $R_2 \rightarrow \overline{H} = R_2 \rightarrow \overline{D}_1 \rightarrow \overline{H}$ ,  $L_1 \rightarrow \overline{H} = L_1 \rightarrow \overline{D}_2 \rightarrow \overline{H}$ ,  $K_1 \rightarrow \overline{D}_1 = K_1 \rightarrow D_0 \rightarrow \overline{D}_1$  und  $R_1 \rightarrow M = R_1 \rightarrow D_2 \rightarrow M$ . Dann folgt mit dem Kompositionslemma für PO's, daß die zusammengesetzten Diagramme (L3)+(L8), (R3)+(R9), (L2)+(L9) und (R2)+(R8) PO's sind. Folglich existiert eine Ableitung  $G \Rightarrow_{r_2} \overline{H} \Rightarrow_{r_1} M$ .

**Ergebnis:** Offensichtlich gibt es Graph-Morphismen  $R_2 \rightarrow \overline{D}_1$  und  $L_1 \rightarrow \overline{D}_2$

$$\begin{array}{ccccc}
L_2 & \leftarrow K_2 & \rightarrow R_2 & & L_1 \leftarrow K_1 \rightarrow R_1 \\
\downarrow & & \downarrow & & \downarrow & \\
G & \leftarrow \overline{D}_2 & \rightarrow \overline{H} & & \leftarrow \overline{D}_1 & \rightarrow M
\end{array}$$

gibt derart, daß  $R_2 \rightarrow \overline{D}_1 \rightarrow \overline{H} = R_2 \rightarrow \overline{H}$  und  $L_1 \rightarrow \overline{D}_2 \rightarrow \overline{H} = L_1 \rightarrow$

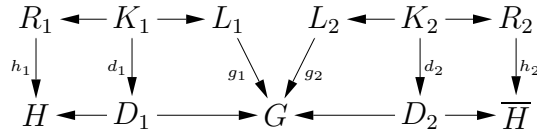
$\overline{H}$  gilt. Somit sind die direkten Ableitungen  $G \Rightarrow_{r_2} \overline{H} \Rightarrow_{r_1} M$  sequentiell unabhängig (Charakterisierung der sequentiellen Unabhängigkeit).  $\square$

### Parallele Unabhängigkeit

Gegeben: Zwei direkte Ableitungen  $G \Rightarrow_{r_1} H$  und  $G \Rightarrow_{r_2} \overline{H}$ .

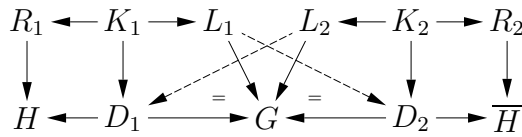
Frage: Gibt es direkte Ableitungen der Form  $H \Rightarrow_{r_2} M \Leftarrow_{r_1} \overline{H}$ ?

**Definition 15. (Parallele Unabhängigkeit)** Zwei direkte Ableitungen  $G \Rightarrow_{r_1, g_1} H$  und  $G \Rightarrow_{r_2, g_2} \overline{H}$  heißen **parallel unabhängig**, wenn  $g_1(L_1) \cap g_2(L_2) \subseteq g_1(K_1) \cap g_2(K_2)$  gilt.



**Beobachtung.** Zwei direkte Ableitungen  $G \Rightarrow_{r_1, g_1} H$  und  $G \Rightarrow_{r_2, g_2} \overline{H}$  sind genau dann parallel unabhängig, wenn die direkten Ableitungen  $H \Rightarrow_{r_1^{-1}, h_1} G$  und  $G \Rightarrow_{r_2, g_2} \overline{H}$  sequentiell unabhängig sind.

**Lemma 20. (Charakterisierung der parallelen Unabhängigkeit)** Zwei direkte Ableitungen  $G \Rightarrow_{r_1, g_1} H$  und  $G \Rightarrow_{r_2, g_2} \overline{H}$  sind genau dann parallel unabhängig, wenn es Graph-Morphismen  $L_1 \rightarrow D_2$  und  $L_2 \rightarrow D_1$  gibt derart, daß  $L_1 \rightarrow D_2 \rightarrow G = L_1 \rightarrow G$  und  $L_2 \rightarrow D_1 \rightarrow G = L_2 \rightarrow G$  gilt.



**Beweis.** (folgt direkt aus der Beobachtung und der Charakterisierung der sequentiellen Unabhängigkeit)  $\square$

**Theorem 21. (parallele Unabhängigkeit)** Seien  $G \Rightarrow_{r_1} H$  und  $G \Rightarrow_{r_2} \overline{H}$  parallel unabhängig. Dann gibt es einen Graphen  $M$  und direkte Ableitungen  $H \Rightarrow_{r_2} M$ ,  $\overline{H} \Rightarrow_{r_1} M$  derart, daß  $G \Rightarrow_{r_1} H \Rightarrow_{r_2} M$  und  $G \Rightarrow_{r_2} \overline{H} \Rightarrow_{r_1} M$  sequentiell unabhängig sind.

**Beweis.** Seien  $G \Rightarrow_{r_1, g_1} H$  und  $G \Rightarrow_{r_2, g_2} \overline{H}$  parallel unabhängig. Dann gibt es Graph-Morphismen  $L_1 \rightarrow D_2$  und  $L_2 \rightarrow D_1$  gibt derart, daß  $L_1 \rightarrow D_2 \rightarrow G = L_1 \rightarrow G$  und  $L_2 \rightarrow D_1 \rightarrow G = L_2 \rightarrow G$  gilt.

**Ausgangssituation:**

$$\begin{array}{ccccccc}
 R_1 & \leftarrow & K_1 & \rightarrow & L_1 & & L_2 & \leftarrow & K_2 & \rightarrow & R_2 \\
 \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\
 H & \leftarrow & D_1 & \xrightarrow{\quad} & G & \xleftarrow{\quad} & D_2 & \rightarrow & \overline{H}
 \end{array}$$

(R0)   (L0)   (L1)   (R1)

**Zerlegung:** Konstruiere nun  $D_0$  als PB-Objekt von  $D_1 \rightarrow G \leftarrow D_2$ . Dann existiert eindeutig ein Graph-Morphismus  $K_1 \rightarrow D_0$  derart, daß  $K_1 \rightarrow D_0 \rightarrow D_1 = K_1 \rightarrow D_1$  und  $K_1 \rightarrow D_0 \rightarrow D_2 = K_1 \rightarrow L_1 \rightarrow D_2$  gilt. Entsprechend existiert eindeutig ein Graph-Morphismus  $K_2 \rightarrow D_0$  derart, daß  $K_2 \rightarrow D_0 \rightarrow D_2 = K_2 \rightarrow D_2$  und  $K_2 \rightarrow D_0 \rightarrow D_1 = K_2 \rightarrow L_2 \rightarrow D_1$  gilt. Konstruiere weiterhin  $\overline{D}_2$  und  $\overline{D}_1$  als PO-Objekte von  $R_1 \leftarrow K_1 \rightarrow D_0$  bzw.  $R_2 \leftarrow K_2 \rightarrow D_0$ . Dann existieren eindeutig Graph-Morphismen  $\overline{D}_2 \rightarrow H$ ,  $\overline{D}_1 \rightarrow \overline{H}$  derart, daß  $R_1 \rightarrow \overline{D}_2 \rightarrow H = R_1 \rightarrow H$  und  $D_0 \rightarrow \overline{D}_2 \rightarrow H = D_0 \rightarrow D_1 \rightarrow H$  bzw.  $R_2 \rightarrow \overline{D}_1 \rightarrow \overline{H} = R_2 \rightarrow \overline{H}$  und  $D_0 \rightarrow \overline{D}_1 \rightarrow \overline{H} = D_0 \rightarrow D_2 \rightarrow \overline{H}$  gilt.

$$\begin{array}{ccccccc}
 R_1 & \leftarrow & K_1 & \rightarrow & L_1 & & L_2 & \leftarrow & K_2 & \rightarrow & R_2 \\
 \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\
 \overline{D}_2 & \leftarrow & D_0 & \xrightarrow{\quad} & D_2 & & D_1 & \leftarrow & D_0 & \xrightarrow{\quad} & \overline{D}_1 \\
 \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\
 H & \leftarrow & D_1 & \xrightarrow{\quad} & G & \xleftarrow{\quad} & D_2 & \rightarrow & \overline{H}
 \end{array}$$

(R2)   (L2)   (L3)   (R3)   (R81)   (L9)   (R82)

Nach Konstruktion ist das Diagramm (L9) ein PB; die Diagramme (R2) und (R3) sind PO's. Im folgenden wollen wir zeigen, daß die Diagramme (L9), (L2), (R81), (L3) und (R82) PO's sind. Da die Ausgangsdiagramme (L0) und (L1) PO's und  $K_1 \rightarrow L_1$  und  $K_2 \rightarrow L_2$  injektiv sind, sind  $(D_1 \rightarrow G, L_1 \rightarrow G)$  und  $(D_2 \rightarrow G, L_2 \rightarrow G)$  gemeinsam surjektiv und  $D_1 \rightarrow G$  und  $D_2 \rightarrow G$  injektiv (PO-Charakterisierung und Injektivitätslemma). Folglich sind  $(D_1 \rightarrow G, D_2 \rightarrow G)$  gemeinsam surjektiv. Da (L9) PB ist, ist (L9) auch PO (PB-PO-Lemma). Da (L0)=(L2)+(L9) und (L9) PO's sind und  $D_2 \rightarrow G$  injektiv ist, ist (L2) PO (spezielles Dekompositionslemma für PO's). Da

(R0)=(R2)+(R81) und (R2) PO's sind, ist (R81) PO (Dekompositionslemma für PO's). Analog folgt, daß die Diagramme (L3) und (R82) PO's sind.

**Zusammensetzung:** Konstruiere nun  $M$  als PO-Objekt zu  $\overline{D}_1 \leftarrow D_0 \rightarrow \overline{D}_2$ .

$$\begin{array}{ccccc}
 L_2 & \xleftarrow{\quad} & K_2 & \xrightarrow{\quad} & R_2 & & R_1 & \xleftarrow{\quad} & K_1 & \xrightarrow{\quad} & L_1 \\
 \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\
 & (L3) & & (R3) & & & (R2) & & (L2) & & \\
 D_1 & \xleftarrow{\quad} & D_0 & \xrightarrow{\quad} & \overline{D}_1 & & \overline{D}_2 & \xleftarrow{\quad} & D_0 & \xrightarrow{\quad} & D_2 \\
 \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\
 & (R81) & & (R9) & & & (R9) & & (R82) & & \\
 H & \xleftarrow{\quad} & \overline{D}_2 & \xrightarrow{\quad} & M & & \overline{D}_1 & \xrightarrow{\quad} & \overline{H} & & 
 \end{array}$$

Mit dem Kompositionslemma für PO's folgt, daß die zusammengesetzten Diagramme (L3)+(R81), (R3)+(R9), (L2)+(R82) und (R2)+(R9) PO's sind. Folglich existieren direkte Ableitungen  $H \Rightarrow_{r_2} M$  und  $\overline{H} \Rightarrow_{r_1} M$ .

**Ergebnis:** Offensichtlich gibt es Graph-Morphismen  $R_1 \rightarrow \overline{D}_2$  und  $L_2 \rightarrow D_1$  gibt derart, daß  $R_1 \rightarrow \overline{D}_2 \rightarrow H = R_1 \rightarrow H$  und  $L_2 \rightarrow D_1 \rightarrow H = L_2 \rightarrow H$  gilt.

$$\begin{array}{ccccc}
 L_2 & \xleftarrow{\quad} & K_2 & \xrightarrow{\quad} & R_2 & & L_1 & \xleftarrow{\quad} & K_1 & \xrightarrow{\quad} & R_1 \\
 \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow & & \downarrow \\
 & & & & & & & & & & \\
 H & \xleftarrow{\quad} & \overline{D}_2 & \xrightarrow{\quad} & M & & \overline{D}_1 & \xrightarrow{\quad} & \overline{H} & & 
 \end{array}$$

Weiterhin gibt es Graph-Morphismen  $R_2 \rightarrow \overline{D}_1$  und  $L_1 \rightarrow D_2$  gibt derart, daß  $R_2 \rightarrow \overline{D}_1 \rightarrow \overline{H} = R_2 \rightarrow \overline{H}$  und  $L_1 \rightarrow D_2 \rightarrow \overline{H} = L_1 \rightarrow \overline{H}$  gilt. Somit sind die direkten Ableitungen  $G \Rightarrow_{r_1} H \Rightarrow_{r_2} M$  und  $G \Rightarrow_{r_2} \overline{H} \Rightarrow_{r_1} M$  sequentiell unabhängig (Charakterisierung der sequentiellen Unabhängigkeit).  $\square$

**Aufgabe 1. (sequentielle und parallele Unabhängigkeit)** Definiere sequentielle und parallele Unabhängigkeit für direkte Ableitungen bzgl. Regeln der Form  $r_i = (L_i \leftarrow K_i \rightarrow R_i)$ , wobei  $K_i \rightarrow L_i$  injektiv und  $K_i \rightarrow R_i$  beliebig ist.

**Aufgabe 2. (sequentielle Unabhängigkeit)** Läßt sich der Satz zur sequentiellen Unabhängigkeit für den allgemeineren Fall, in dem die Regeln



von der Form  $r_i = (L_i \leftarrow K_i \rightarrow R_i)$  (mit  $K_i \rightarrow L_i$  injektiv) sind, beweisen? Wenn nein, woran liegt das? Welche Aussage kann man dennoch treffen?

**Aufgabe 3. (parallele Unabhängigkeit)** Läßt sich der Satz zur parallelen Unabhängigkeit auch für den allgemeineren Fall, in dem die Regeln von der Form  $r_i = (L_i \leftarrow K_i \rightarrow R_i)$  (mit  $K_i \rightarrow L_i$  injektiv) sind, beweisen? Wenn ja, woran liegt das?

## 6 Parallelableitungen

**Definition 16. (Parallelregel)** Seien  $r_1 = (L_1 \supseteq K_1 \subseteq R_1)$  und  $r_2 = (L_2 \supseteq K_2 \subseteq R_2)$  zwei Regeln. Dann heit  $r_1 + r_2 = (L_1 + L_2 \supseteq K_1 + K_2 \subseteq R_1 + R_2)$  die **Parallelregel** von  $r_1$  und  $r_2$ . Fr eine Regelmengende  $\mathcal{R}$  ist der **Parallelabschlu**  $\mathcal{R}^+$  induktiv definiert durch:

- (1)  $\mathcal{R} \subseteq \mathcal{R}^+$  und
- (2) fr  $r_1, r_2 \in \mathcal{R}^+$  ist auch  $r_1 + r_2 \in \mathcal{R}^+$ .

Eine Ableitung  $G \Rightarrow_{\mathcal{R}^+}^* H$  heit **Parallelableitung**.

**Bemerkung.**

1.  $\mathcal{R}^+$  ist i.a. unendlich.
2. Fr  $r_1, r_2, r_3 \in \mathcal{R}^+$  gilt:  $r_1 + r_2 = r_2 + r_1$  und  $r_1 + (r_2 + r_3) = (r_1 + r_2) + r_3$  (bis auf Isomorphie).

**Beobachtung.**  $G \Rightarrow_{\mathcal{R}}^* H$  impliziert  $G \Rightarrow_{\mathcal{R}^+}^* H$ .

**Theorem 22. (Sequentialisierungssatz)** Sei  $G \Rightarrow_{r_1+r_2} M$  eine direkte Parallelableitung. Dann gibt es zwei Graphen  $H$  und  $\overline{H}$  sowie sequentiell unabhngige Ableitungen  $G \Rightarrow_{r_1} H \Rightarrow_{r_2} M$  und  $G \Rightarrow_{r_2} \overline{H} \Rightarrow_{r_1} M$ . (Die entstehenden Ableitungen  $G \Rightarrow_{r_1} H \Rightarrow_{r_2} M$  und  $G \Rightarrow_{r_2} \overline{H} \Rightarrow_{r_1} M$  heien **Sequentialisierungen** der Parallelableitung  $G \Rightarrow_{r_1+r_2} M$ ).

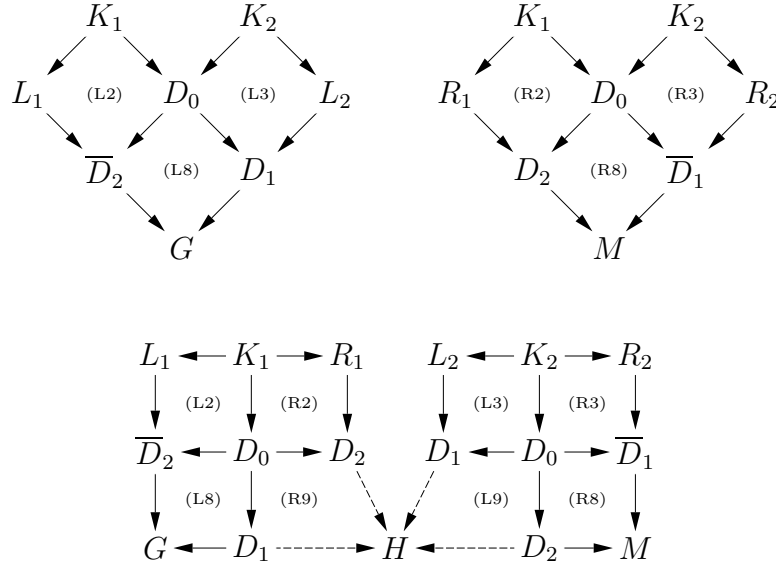
**Beweis.**

**Ausgangssituation:**

$$\begin{array}{ccccc}
 L_1 + L_2 & \longleftarrow & K_1 + K_2 & \longrightarrow & R_1 + R_2 \\
 \downarrow & & \downarrow & & \downarrow \\
 & \text{(L1)} & & \text{(R1)} & \\
 G & \longleftarrow & D_0 & \longrightarrow & M
 \end{array}$$

**Zerlegung von (L1) und (R1):**

Nach dem Schmetterlingslemma existieren Zerlegungen der PO-Diagramme (L1) und (R1) in PO-Diagramme (L2), (L3), (L8), und (R2), (R3), (R8).

**Zusammensetzung:**

Konstruiere nun  $H$  als PO-Objekt von  $D_1 \leftarrow D_0 \rightarrow D_2$ . **Ergebnis:** Da  $R_1 \rightarrow H = R_1 \rightarrow D_2 \rightarrow H$  und  $L_2 \rightarrow H = L_2 \rightarrow D_1 \rightarrow H$ , sind die direkten Ableitungen  $G \Rightarrow_{r_1} H \Rightarrow_{r_2} M$  sequentiell unabhängig (Charakterisierung der sequentiellen Unabhängigkeit).

Entsprechend lassen sich sequentiell unabhängige direkten Ableitungen  $G \Rightarrow_{r_2} \bar{H} \Rightarrow_{r_1} M$  konstruieren.  $\square$

**Folgerung.**  $G \Rightarrow_{\mathcal{R}}^* H$  genau dann, wenn  $G \Rightarrow_{\mathcal{R}^+}^* H$ .

(Parallelität erhöht nicht die Mächtigkeit, verkürzt aber Ableitungen.)

**Theorem 23. (Parallelisierungssatz)** Seien  $G \Rightarrow_{r_1} H \Rightarrow_{r_2} M$  zwei sequentiell unabhängige direkte Ableitungen. Dann gibt es eine direkte Parallelableitung  $G \Rightarrow_{r_1+r_2} M$ .

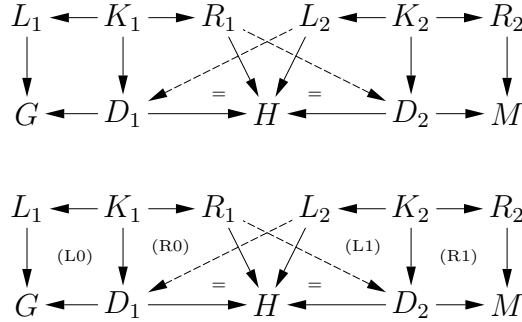
(Die entstehende Ableitung  $G \Rightarrow_{r_1+r_2} M$  heißt **Parallelisierung** der Ableitung  $G \Rightarrow_{r_1} H \Rightarrow_{r_2} M$ ).

**Beweis.**

**Ausgangssituation:**

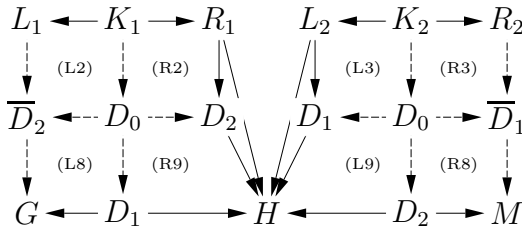
**Zerlegung:**

Konstruiere nun  $D_0$  als PB-Objekt von  $D_1 \rightarrow H \leftarrow D_2$ . Dann existiert



eindeutig ein Graph-Morphismus  $K_1 \rightarrow D_0$  derart, daß  $K_1 \rightarrow D_0 \rightarrow D_1 = K_1 \rightarrow D_1$  und  $K_1 \rightarrow D_0 \rightarrow D_2 = K_1 \rightarrow R_2 \rightarrow D_2$  gilt. Entsprechend existiert eindeutig ein Graph-Morphismus  $K_2 \rightarrow D_0$  derart, daß  $K_2 \rightarrow D_0 \rightarrow D_2 = K_2 \rightarrow D_2$  und  $K_2 \rightarrow D_0 \rightarrow D_1 = K_2 \rightarrow L_2 \rightarrow D_1$  gilt.

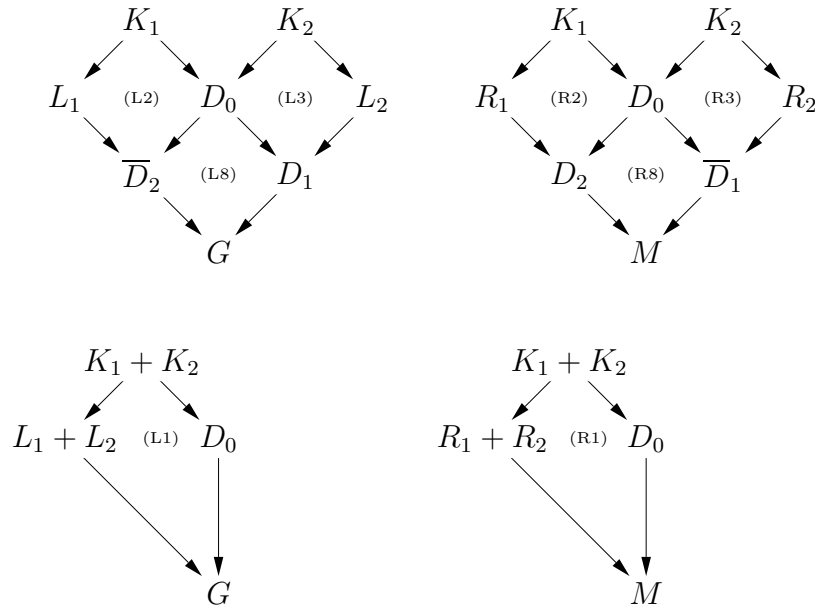
Konstruiere weiterhin  $\overline{D}_2$  und  $\overline{D}_1$  als PO-Objekte von  $L_1 \leftarrow K_1 \rightarrow D_0$  bzw.  $R_2 \leftarrow K_2 \rightarrow D_0$ . Dann existieren eindeutig Graph-Morphismen  $\overline{D}_2 \rightarrow G$ ,  $\overline{D}_1 \rightarrow M$  derart, daß  $L_1 \rightarrow \overline{D}_2 \rightarrow G = L_1 \rightarrow G$  und  $D_0 \rightarrow \overline{D}_2 \rightarrow G = D_0 \rightarrow D_1 \rightarrow G$  bzw.  $R_2 \rightarrow \overline{D}_1 \rightarrow M = R_2 \rightarrow M$  und  $D_0 \rightarrow \overline{D}_1 \rightarrow M = D_0 \rightarrow D_2 \rightarrow M$  gilt. Die entstehenden Diagramme sind PO's (s.o).



**Anwenden des Schmetterlingslemmas:** Nach dem Schmetterlingslemma sind (L1) und (R1) PO-Diagramme. Folglich existiert eine direkte Parallelableitung  $G \Rightarrow_{r_1+r_2} M$ .  $\square$

**Folgerung.** Seien die direkten Ableitungen  $G \Rightarrow_{r_1} H$  und  $G \Rightarrow_{r_2} \overline{H}$  parallel unabhängig. Dann gibt es einen Graphen  $M$ , direkte Ableitungen  $H \Rightarrow_{r_2} M$  und  $\overline{H} \Rightarrow_{r_1} M$  und eine Parallelableitung  $G \Rightarrow_{r_1+r_2} M$ .

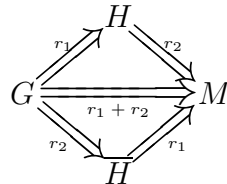
**Aufgabe 4. (Parallelregel)** Definiere die Parallelregel  $r_1 + r_2$  von Regeln der Form  $r_i = (L_i \leftarrow K_i \rightarrow R_i)$  ( $i = 1, 2$ ), wobei  $K_i \rightarrow L_i$  injektiv und



$K_i \rightarrow R_i$  beliebig ist.

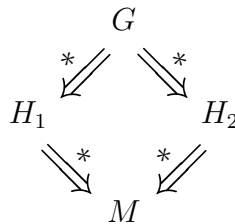
**Aufgabe 5. (Sequentialisierungssatz)** Läßt sich der Sequentialisierungssatz für den allgemeineren Fall beweisen? Wenn ja, woran liegt das?

**Aufgabe 6. (Parallelisierungssatz)** Läßt sich der Parallelisierungssatz für den allgemeineren Fall beweisen? Wenn nein, woran liegt das?



## 7 Konfluenz

**Definition 17. (Konfluenz)** Ein Graphersetzungs-System  $\mathcal{G}$  (bzw. die dazugehörige Ersetzungsrelation  $\Rightarrow$ ) ist **konfluent**, wenn es für alle Graphen  $G, H_1, H_2$  mit  $H_1 \Leftarrow^* G \Rightarrow^* H_2$  einen Graphen  $M$  mit  $H_1 \Rightarrow^* M \Leftarrow^* H_2$  gibt.



**Definition 18. (Normalform)** Ein Graph  $G$  ist in **Normalform** bzgl.  $\mathcal{G}$  (bzw.  $\Rightarrow$ ), wenn es keinen Graphen  $H$  mit  $G \Rightarrow H$  gibt. Ist  $\overline{G} \Rightarrow^* G$  eine Ableitung und  $G$  in Normalform, so ist  $G$  eine **Normalform von  $\overline{G}$** . Graphen in Normalform werden auch **reduzierte Graphen** genannt.

### Eindeutigkeit der Normalform

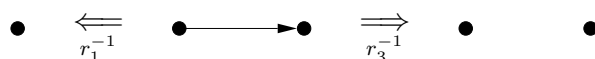
**Lemma 24. (Eindeutigkeit der Normalform)** Wenn  $\mathcal{G}$  konfluent ist, dann besitzt jeder Graph (bis auf Isomorphie) höchstens eine Normalform.

**Beweis.** Seien  $H_1$  und  $H_2$  Normalformen eines Graphen  $G$ . Dann gilt  $H_1 \Leftarrow^* G \Rightarrow^* H_2$ . Wegen der Konfluenz gibt es einen Graphen  $M$  mit  $H_1 \Rightarrow^* M \Leftarrow^* H_2$ . Da  $H_1$  und  $H_2$  Normalformen sind, haben die Ableitungen  $H_1 \Rightarrow^* M$  und  $H_2 \Rightarrow^* M$  die Länge 0, d.h.  $H_1 \cong M \cong H_2$ .  $\square$

### Beispiel 13. (Konfluenz)

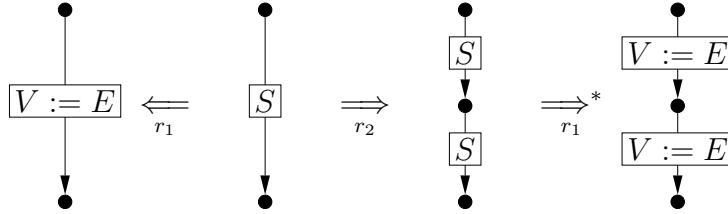
CON ist konfluent: Es existieren keine kritischen Paare.

CON<sup>-1</sup> ist nicht konfluent:



$\text{WSF}^{-1}$  ist konfluent: Alle kritischen Paare sind stark zusammenführbar.

WSF ist nicht konfluent:



**Bemerkung.** Graphersetzungssysteme, bei denen jeder Graph höchstens eine Normalform hat, müssen nicht konfluent sein: Betrachte z.B. das Graphersetzungssystem  $\mathcal{G}$  mit der Regelmeng

$$\mathcal{R} \begin{cases} r_1 = \langle @ \supseteq \emptyset \subseteq @ \rangle \\ r_2 = \langle @ \supseteq \emptyset \subseteq c \rangle \\ r_3 = \langle c \supseteq c \subseteq c \rangle \end{cases}$$

Dann hat jeder Graph eine Normalform (der Graph, in dem alle isolierten Knoten mit Markierung  $a$  durch einen Knoten mit Markierung  $b$  ersetzt sind). Das Graphersetzungssystem ist jedoch nicht konfluent:

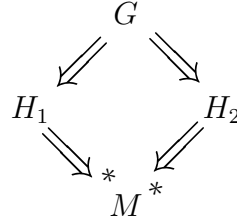
$$\begin{array}{c} @ \xRightarrow[r_2]{r_1} c \xRightarrow[r_3]{r_2} c \xRightarrow[r_3]{r_3} c \xRightarrow[r_3]{r_3} c \dots \\ \Downarrow r_1 \\ b \end{array}$$

**Lemma 25. (Konfuenz)**  $\mathcal{G}$  ist genau dann konfluent, wenn es für alle Graphen  $G, H_1, H_2$  mit  $H_1 \leftarrow G \Rightarrow^* H_2$  einen Graphen  $M$  mit  $H_1 \Rightarrow^* M \Leftarrow^* H_2$  gibt.

**Beweis.** Durch Induktion über  $n$ , wobei  $G \Rightarrow^n H_2$ . □

**Definition 19. (lokale Konfuenz & Termination)** Ein Graphersetzungssystem  $\mathcal{G}$  (bzw. die dazugehörige Ersetzungsrelation  $\Rightarrow$ ) ist **lokal konfluent**, wenn es für alle Graphen  $G, H_1, H_2$  mit  $H_1 \leftarrow G \Rightarrow H_2$  einen

Graphen  $M$  mit  $H_1 \Rightarrow^* M \Leftarrow^* H_2$  gibt.

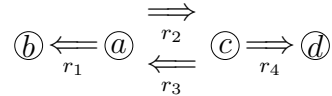


$\mathcal{G}$  (bzw.  $\Rightarrow$ ) ist **terminierend** (oder **noethersch**), wenn es keine unendliche Folge  $G_0 \Rightarrow G_1 \Rightarrow G_2 \Rightarrow \dots$  gibt.

**Bemerkung.** Konfluenz impliziert lokale Konfluenz. Die Umkehrung gilt i.a. jedoch nicht. Betrachte z.B. das Graphersetzungssystem  $\mathcal{G}$  mit der Regelmeng

$$\mathcal{R} \begin{cases} r_1 = \langle @ \supseteq \emptyset \subseteq b \rangle \\ r_2 = \langle @ \supseteq \emptyset \subseteq c \rangle \\ r_3 = \langle c \supseteq \emptyset \subseteq @ \rangle \\ r_4 = \langle c \supseteq \emptyset \subseteq d \rangle \end{cases}$$

Dann ist  $\mathcal{G}$  lokal konfluent, jedoch nicht konfluent:



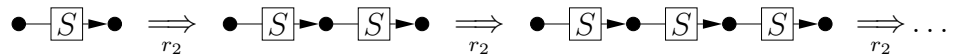
Beachte, daß  $\mathcal{G}$  ein nichtterminierendes System ist.

#### Beispiel 14. (Termination)

- CON ist nicht terminierend:



- WSF ist nicht terminierend:





- $\text{CON}^{-1}$  ist terminierend: Für jeden Schritt  $G \Rightarrow_{\text{CON}^{-1}} H$  gilt  $|E_G| > |E_H|$  und für jeden Graphen  $M$  ist die Kantenzahl  $|E_M| \geq 0$ .
- $\text{WSF}^{-1}$  ist terminierend: Sei  $\#G = |E_G| + |\{e \in E_G \mid m_G(e) = V := E\}|$ . Für jeden Schritt  $G \Rightarrow_{\text{WSF}^{-1}} H$  gilt  $\#G > \#H$  und für jeden Graphen  $M$  ist  $\#M \geq 0$ .

### Noethersche Induktion für Graphersetzung

Sei  $\Rightarrow$  terminierend (noethersch) und  $P$  eine  $\Rightarrow$ -vollständige Eigenschaft auf Graphen, d.h. für alle Graphen  $G$  gilt:  $P(H)$  für alle  $H$  mit  $G \Rightarrow^+ H$  impliziert  $P(G)$ . Dann gilt  $P(G)$  für alle Graphen  $G$ . Dabei steht  $\Rightarrow^+$  für Ableitungen der Länge  $\geq 1$ .

**Theorem 26. (Newman's Lemma für Graphersetzungssysteme)** Ein terminierendes Graphersetzungssystem ist genau dann konfluent, wenn es lokal konfluent ist.

**Beweis.** " $\Rightarrow$ ": trivial.

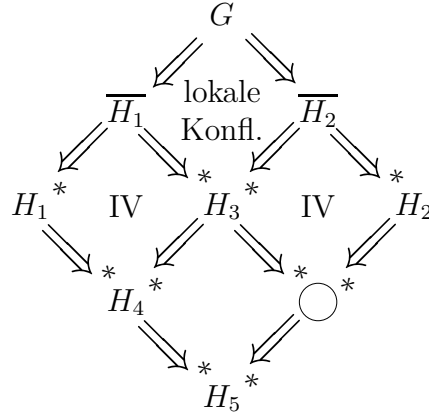
" $\Leftarrow$ ": Sei  $\Rightarrow$  terminierend und lokal konfluent. Wir zeigen

$$P(G): \text{Für alle } H_1, H_2 \text{ mit } H_1 \Leftarrow^* G \Rightarrow^* H_2 \exists M : H_1 \Rightarrow^* M \Leftarrow^* H_2$$

durch noethersche Induktion, indem wir zeigen, daß  $P$   $\Rightarrow$ -vollständig ist. Seien  $G \Rightarrow^m H_1$  und  $G \Rightarrow^n H_2$  Ableitungen. Wir zeigen:  $\exists M : H_1 \Rightarrow^* M \Leftarrow^* H_2$ .

1. Falls  $m = 0$ , wähle  $M = H_2$ ; falls  $n = 0$ , wähle  $M = H_1$ .
2. Anderenfalls sei  $G \Rightarrow \overline{H_1} \Rightarrow^* H_1$  und  $G \Rightarrow \overline{H_2} \Rightarrow^* H_2$ . Wegen der lokalen Konfluenz  $\exists H_3 : \overline{H_1} \Rightarrow^* H_3 \Leftarrow^* \overline{H_2}$ . Nach Induktionsvoraussetzung ist  $P(\overline{H_1})$ , also  $\exists H_4 : \overline{H_1} \Rightarrow^* H_4 \Leftarrow^* H_3$ . Nach Induktionsvoraussetzung ist  $P(\overline{H_2})$ , also  $\exists H_5 : H_3 \Rightarrow^* H_5 \Leftarrow^* \overline{H_2}$ . Damit gilt  $P(G)$ .

Der Induktionsschritt des Beweises ist in folgendem Diagramm gezeigt:



□

### Überprüfung der lokalen Konfluenz

**Frage:** Wie überprüft man Graphersetzungssysteme auf lokale Konfluenz?

**Idee:** Überprüfe nur solche Schritte  $T \leftarrow_{r_1} S \Rightarrow_{r_2} U$ , bei denen  $S$  durch “kritische Überlagerung” der linken Seiten von  $r_1$  und  $r_2$  entsteht.

**Definition 20. (Kritischen Paare)** Seien  $r_i = (L_i \supseteq K_i \subseteq R_i)$  Regeln für  $i = 1, 2$ . Zwei direkte Ableitungen  $T \leftarrow_{r_1, g_1} S \Rightarrow_{r_2, g_2} U$  bilden ein **kritisches Paar**, falls  $S = g_1(L_1) \cup g_2(L_2)$  ist und  $g_1(L_1) \cap g_2(L_2) \neq g_1(K_1) \cap g_2(K_2)$ . Außerdem soll  $g_1 \neq g_2$  im Fall  $r_1 = r_2$  gelten. Ein kritisches Paar  $T \leftarrow S \Rightarrow U$  ist **zusammenführbar**, wenn es einen Graphen  $X$  mit  $T \Rightarrow^* X \Leftarrow^* U$  gibt.

**Bemerkung.** Graphersetzungssysteme mit einer (bis auf Isomorphie) endlichen Regelmengge besitzen nur endlich viele kritische Paare.

**Theorem 27. (keine kritischen Paare)** Sei  $\mathcal{G}$  ein Graphersetzungssystem ohne kritische Paare. Dann gilt: Für alle Graphen  $G, H_1, H_2$  mit  $H_1 \Leftarrow G \Rightarrow H_2$  gilt:  $H_1 \cong H_2$  oder es gibt einen Graphen  $M$  mit  $H_1 \Rightarrow M \Leftarrow H_2$ .

**Beweis.** Sei  $H_1 \leftarrow_{r_1, g_1} G \Rightarrow_{r_2, g_2} H_2$ . Ist  $g_1(L_1) \cap g_2(L_2) = g_1(K_1) \cap g_2(K_2)$ , so sind die direkten Ableitungen  $H_1 \Leftarrow G \Rightarrow H_2$  parallel unabhängig und es gibt

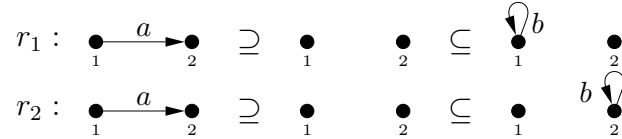
einen Graphen  $M$  mit  $H_1 \Rightarrow M \Leftarrow H_2$ . Ist  $g_1(L_1) \cap g_2(L_2) \neq g_1(K_1) \cap g_2(K_2)$ , so gibt es nach dem Einschränkunglemma Ableitungen  $T \Leftarrow_{r_1, g'_1} g_1(L_1) \cup g_2(L_2) \Rightarrow_{r_2, g'_2} U$  mit  $g'_1(L_1) \cap g'_2(L_2) \neq g'_1(K_1) \cap g'_2(K_2)$ . Da keine kritischen Paare existieren, muß  $r_1 = r_2$  und  $g_1 = g_2$  sein. Da Ableitungen bis auf Isomorphie eindeutig sind, ist  $H_1 \cong H_2$ .  $\square$

**Folgerung.** Graphersetzungssysteme ohne kritische Paare sind konfluent.

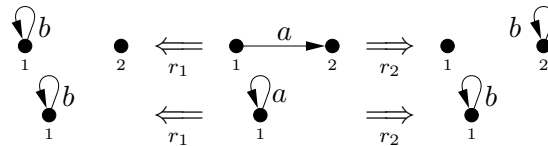
**Beispiel 15.** CON besitzt keine kritischen Paare, weil keine Regel Knoten oder Kanten löscht. Für jede Regel  $(L \supseteq K \subseteq R)$  in CON ist  $L = K$ . Somit ist CON konfluent.

**Bemerkung.** Ein **Termersetzungssystem** ist lokal konfluent, wenn seine kritischen Paare zusammenführbar sind. Für Graphersetzungssysteme folgt aus der Zusammenführbarkeit aller kritischen Paare i.a. noch nicht die lokale Konfluenz (siehe Huet 80 [Hue80]).

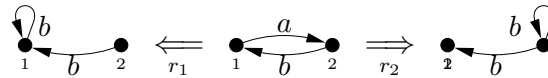
Betrachte das Graphersetzungssystem  $\mathcal{G}$  mit der Regelmeng



Es gibt zwei kritische Paare, die beide zusammenführbar sind:



Das Graphersetzungssystem  $\mathcal{G}$  ist nicht lokal konfluent:



**Definition 21. (track-Funktion)** Sei  $G \Rightarrow H$  eine direkte Ableitung mit dem Diagramm

$$\begin{array}{ccccc} L & \supseteq & K & \subseteq & R \\ g \downarrow & & d \downarrow & & \downarrow h \\ G & \supseteq & D & \subseteq & H \xrightarrow{i} M \end{array}$$

Dann ist die partielle Abbildung  $\text{track}_{G \Rightarrow H}: V_G \rightarrow V_H$  definiert durch

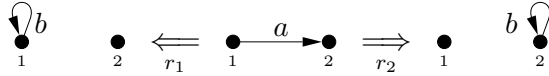
$$\text{track}_{G \Rightarrow H}(v) = \begin{cases} i_V(v) & \text{falls } v \in V_D \\ \text{undefiniert} & \text{sonst} \end{cases}$$

Für eine Ableitung  $G \Rightarrow^* H$  ist  $\text{track}_{G \Rightarrow^* H}: V_G \rightarrow V_H$  definiert durch

$$\text{track}_{G \Rightarrow^* H}(v) = \begin{cases} i_V(v) & \text{falls } G \Rightarrow^* H \text{ durch Isomorphismus } i: G \rightarrow H \text{ gegeben} \\ \text{track}_{G_{n-1} \Rightarrow H}(\dots \text{track}_{G_1 \Rightarrow G_2}(\text{track}_{G \Rightarrow G_1}(v)) \dots) & \text{falls } G \Rightarrow^* H = G \Rightarrow G_1 \Rightarrow \dots \Rightarrow G_{n-1} \Rightarrow H \end{cases}$$

**Definition 22. (Starke Zusammenführbarkeit)** Sei  $T \Leftarrow S \Rightarrow U$  ein kritische Paar und sei  $\text{Protect}(S)$  die Menge aller Knoten in  $S$ , so daß  $\text{track}_{S \Rightarrow T}(v)$  und  $\text{track}_{S \Rightarrow U}(v)$  definiert sind.  $T \Leftarrow S \Rightarrow U$  ist **stark zusammenführbar**, wenn es einen Graphen  $X$  und Ableitungen  $T \Rightarrow^* X \Leftarrow^* U$  gibt, so daß  $\text{track}_{S \Rightarrow T \Rightarrow^* X}(v)$  und  $\text{track}_{S \Rightarrow U \Rightarrow^* X}(v)$  für jedes  $v$  aus  $\text{Protect}(S)$  definiert und gleich sind.

**Beispiel 16.** Das kritische Paar



ist zusammenführbar, aber nicht stark zusammenführbar.

**Theorem 28. (Kritisches-Paar-Lemma für Graphersetzung)** Ein Graphersetzungssystem ist lokal konfluent, wenn seine kritischen Paare stark zusammenführbar sind.

**Beweis.** Seien alle kritischen Paare von  $\mathcal{G}$  streng zusammenführend. Betrachte zwei direkte Ableitungen  $H_1 \Leftarrow_{r_1, g_1} G \Rightarrow_{r_2, g_2} H_2$  bzgl. der Regeln  $r_i = (L_i \supseteq K_i \subseteq R_i)$ ,  $i = 1, 2$ . Falls  $g_1(L_1) \cap g_2(L_2) = g_1(K_1) \cap g_2(K_2)$  ist, so sind die direkten Ableitungen  $H_1 \Leftarrow G \Rightarrow H_2$  parallel unabhängig und es gibt einen Graphen  $M$  mit  $H_1 \Rightarrow M \Leftarrow H_2$ . Setze deshalb  $g_1(L_1) \cap g_2(L_2) \neq g_1(K_1) \cap g_2(K_2)$  voraus. Setze ferner voraus, daß  $r_1 \neq r_2$  oder  $g_1 \neq g_2$  ist, da anderenfalls  $H_1 \cong H_2$  gilt. Sei  $S = g_1(L_1) \cup g_2(L_2)$ . Nach dem Einschränkunglemma gibt es Ableitungen  $U_1 \Leftarrow_{r_1, g'_1} S \Rightarrow_{r_2, g'_2} U_2$ , wobei für  $i = 1, 2$   $g'_i$  die Einschränkung von  $g_i$  auf  $S$  und  $U_i \subseteq H_i$  ist. Offensichtlich bilden diese direkten Ableitungen ein kritisches Paar. Nach Voraussetzung gibt es Ableitungen  $U_1 \Rightarrow^* X \Leftarrow^* U_2$ , so daß  $\text{track}_{S \Rightarrow U_1 \Rightarrow^* X}(v)$  und  $\text{track}_{S \Rightarrow U_2 \Rightarrow^* X}(v)$  für jedes  $v$  aus  $\text{Protect}(S)$  definiert und gleich sind.

Betrachte nun die Ableitungen  $S \Rightarrow_{r_i, g'_i} U_i \Rightarrow^* X$ . Sei  $\text{Boundary}$  der diskrete Teilgraph von  $S$ , der aus allen Knoten besteht, die durch eine Kante in  $G - S$  berührt werden. Dann sind  $\text{track}_{G \Rightarrow H_1}$  und  $\text{track}_{G \Rightarrow H_2}$  für alle Knoten in  $\text{Boundary}$  definiert, da  $G \Rightarrow H_1$  und  $G \Rightarrow H_2$  die Kontaktbedingung erfüllen. Insbesondere sind  $\text{track}_{S \Rightarrow U_1}$  und  $\text{track}_{S \Rightarrow U_2}$  auf  $\text{Boundary}$  definiert, d.h.

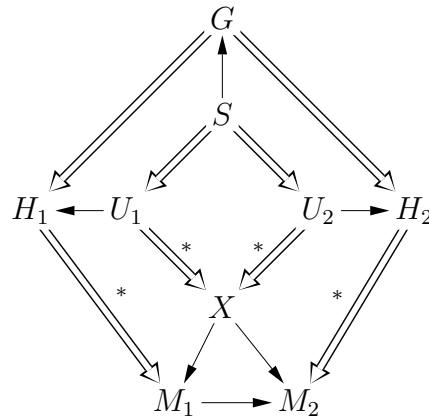
$$\text{Boundary} \subseteq \text{Protect}(S).$$

Deshalb sind für  $i = 1, 2$  auch  $\text{track}_{S \Rightarrow U_i \Rightarrow^* X}$  auf  $\text{Boundary}$  definiert.

Nach dem Einbettungslemma existieren Ableitungen  $G \Rightarrow_{r_i, \overline{g_i}} \overline{H_i} \Rightarrow^* M_i$  mit  $U_i \subseteq \overline{H_i}$  für  $i = 1, 2$ . Darüberhinaus besagt das Einbettungslemma, daß für  $i = 1, 2$ ,  $M_i$  als Verklebungsobjekt von  $\text{Context}$  und  $X$  bzgl.  $\text{Boundary} \rightarrow \text{Context}$  konstruiert werden kann, wobei  $\text{Context} = G - (S - \text{Boundary})$ ,  $\text{Boundary} \rightarrow \text{Context}$  die Inklusion von  $\text{Boundary}$  in  $\text{Context}$  und  $tr_i$  die Einschränkung von  $\text{track}_{S \Rightarrow U_i \Rightarrow^* X}$  auf  $\text{Boundary}$  ist.

$$\begin{array}{ccccc} S & \xleftarrow{\quad} & \text{Boundary} & \xrightarrow{tr_i} & X \\ \downarrow & & \downarrow & & \downarrow \\ G & \xleftarrow{\quad} & \text{Context} & \longrightarrow & M_i \end{array}$$

$tr_1 = tr_2$  impliziert, daß  $M_1 \cong M_2$  ist, da Verklebungsobjekte bis auf Isomorphie eindeutig sind. Folglich gilt  $H_1 \Rightarrow^* M_1 \Leftarrow^* H_2$ .



□

**Folgerung.** Ein terminierendes Graphersetzungssystem ist konfluent, wenn seine kritischen Paare stark zusammenführbar sind.

**Beispiel 17.**  $\text{WSF}^{-1}$  ist terminierend und alle kritischen Paare sind stark zusammenführbar (siehe Übung). Somit ist  $\text{WSF}^{-1}$  konfluent.

**Bemerkung.** Die Umkehrung der Folgerung gilt i.a. nicht: Es gibt Graphersetzungssysteme, die terminierend und konfluent sind, für die jedoch nicht alle kritischen Paare stark zusammenführbar sind. Betrachte z.B. das Graphersetzungssystem  $\mathcal{G}$  mit der Regel

$$r = \begin{array}{c} \bullet \\ \downarrow \\ 1 \end{array} \quad \begin{array}{c} \bullet \\ \longrightarrow \\ 2 \end{array} \quad \begin{array}{c} \bullet \\ \longrightarrow \\ 3 \end{array} \supseteq \begin{array}{c} \bullet \\ \downarrow \\ 1 \end{array} \quad \begin{array}{c} \bullet \\ \longrightarrow \\ 2 \end{array} \quad \begin{array}{c} \bullet \\ \longrightarrow \\ 3 \end{array} \subseteq \begin{array}{c} \bullet \\ \downarrow \\ 1 \end{array} \quad \begin{array}{c} \bullet \\ \longrightarrow \\ 2 \end{array} \quad \begin{array}{c} \bullet \\ \longrightarrow \\ 3 \end{array}$$

- $\mathcal{G}$  ist terminierend, da jeder Schritt  $G \Rightarrow_r H$  die Kantenanzahl verringert.
- $\mathcal{G}$  ist konfluent: Seien Ableitungen  $H_1 \Leftarrow^* G \Rightarrow^* H_2$  gegeben. Falls  $G$  keine Schlinge enthält, gilt  $H_1 \cong G \cong H_2$ . Anderenfalls enthalten  $G$ ,  $H_1$  und  $H_2$  mindestens eine Schlinge und gleich viele Knoten. Dann gilt  $H_1 \Rightarrow^* M \Leftarrow^* H_2$ , wobei  $M$  die gleiche Knotenzahl wie  $G$  hat, eine Schlinge besitzt und keine weiteren Kanten enthält.
- Es gibt kritische Paare, die nicht stark zusammenführbar sind:

$$\begin{array}{c} \bullet \\ \downarrow \\ 1 \end{array} \quad \begin{array}{c} \bullet \\ \longrightarrow \\ 2 \end{array} \Leftarrow_r \begin{array}{c} \bullet \\ \downarrow \\ 1 \end{array} \quad \begin{array}{c} \bullet \\ \downarrow \\ 2 \end{array} \Rightarrow_r \begin{array}{c} \bullet \\ \longrightarrow \\ 1 \end{array} \quad \begin{array}{c} \bullet \\ \downarrow \\ 2 \end{array}$$

**Lemma 29.** Für terminierende Graphersetzungssysteme mit endlich vielen Regeln ist entscheidbar, ob alle kritischen Paare stark zusammenführbar sind.

**Beweis.** siehe Übung □

**Bemerkung.** Für terminierende Termersetzungssysteme ist Konfluenz entscheidbar (siehe z.B. Knuth-Bendix 70 [KB70]).

**Theorem 30. (Unentscheidbarkeit von Konfluenz)** Für terminierende Graphersetzungssysteme mit endlich vielen Regeln ist Konfluenz unentscheidbar.

**Beweis.** (durch Reduktion des Postschen Korrespondenzproblems) Für jede Instanz  $(A, B)$  des PKP wird ein terminierendes Graphersetzungssystem  $\mathcal{G}(A, B)$  konstruiert, das genau dann konfluent ist, wenn  $(A, B)$  keine Lösung hat (siehe Plump 93 [Plu93]).  $\square$

**Aufgabe 7.** Sei  $\mathcal{G}$  ein Graphersetzungssystem und  $\Rightarrow$  die dazugehörige Ersetzungsrelation.

- a) Zeige, daß  $\Rightarrow$  genau dann konfluent ist, wenn es für alle Graphen  $G, H_1, H_2$  mit  $H_1 \Leftarrow G \Rightarrow^* H_2$  einen Graphen  $M$  mit  $H_1 \Rightarrow^* M \Leftarrow^* H_2$  gibt. (Hinweis: Verwende Induktion über die Länge von Ableitungen.)
- b)  $\mathcal{G}$  habe die Eigenschaft, daß für alle Graphen  $G, H_1, H_2$  mit  $H_1 \Rightarrow G \Rightarrow H_2$  gilt:  $H_1 \cong H_2$  oder es gibt einen Graphen  $M$  mit  $H_1 \Rightarrow M \Leftarrow H_2$ . Zeige, daß diese Eigenschaft Konfluenz impliziert. (Hinweis: Verwende Induktion über die Länge von Ableitungen und a).)

**Aufgabe 8.** (Kritische Paare) Sei  $\text{WSF}^{-1} = \{r_1^{-1}, \dots, r_4^{-1}\}$  die Menge der Umkehrregeln für die Regeln zur Erzeugung wohlstrukturierter Flußdiagramme.

- a) Bestimme alle kritischen Paare.
- b) Zeige, daß alle kritischen Paare stark zusammenführbar sind.

## Literatur

- [Hue80] Gérard Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the ACM* 27(4), 797–821, 1980.
- [KB70] Donald E. Knuth, Peter B. Bendix. Simple word problems in universal algebras. In J. Leech, ed., *Computational Problems in Abstract Algebras*, 263–297. Pergamon Press, 1970.
- [Plu93] Detlef Plump. Hypergraph rewriting: Critical pairs and undecidability of confluence. In Ronan Sleep, Rinus Plasmeijer, Marko van Eekelen, eds., *Term Graph Rewriting: Theory and Practice*. John Wiley, New York, 1993.

## 8 Graphprogramme

Existing languages

PROGRES
AGG
GAMMA
GRR
DACTL
⋮

Wanted: core language

- |   |
|---|
| (1) nondeterministic one-step application of a set of rules |
| (2) sequential composition                                  |
| (3) iteration in form of <b>apply as long as possible</b>   |

- many concepts for controlling rules
- **universal power?**
- **semantics?**

- simple/minimal
- universal power
- formal semantics

What control structures are needed for computational completeness?

### Graph transformation

**Graphs:** Directed, labelled (nodes and edges), multiple edges allowed

**Rules:**  $r = (L \leftarrow K \rightarrow R)$  where  $L \leftarrow K$  is injective

### Rule application

- (1) Find injective graph morphism  $g: L \rightarrow G$  satisfying the **dangling condition**: no edge in  $G - g(L)$  is incident to a node in  $g(L) - g(K)$
- (2) Remove  $g(L - K)$  from  $G$  to obtain  $D$
- (3) Glue together  $D$  and  $R$  in  $K$  to obtain  $H$  (pushout construction)

$$\begin{array}{ccccc}
 L & \leftarrow & K & \rightarrow & R \\
 \downarrow & & \downarrow & & \downarrow \\
 G & \leftarrow & D & \rightarrow & H
 \end{array}$$

Notation:  $G \Rightarrow_r H$



## 8.1 Programs

### Definition 23. (Program)

- (1) Every finite set  $\mathcal{R}$  of rules is a program [**elementary programs**]
- (2) If  $P_1$  and  $P_2$  are programs, then  $\langle P_1; P_2 \rangle$  is a program [**sequential composition**]
- (3) If  $P$  is a program according to (1) or (2), then  $P\downarrow$  is a program [**iteration**]

### Semantics

Semantics of  $P$  is a binary relation  $\rightarrow_P$  on (abstract) graphs:

- (1)  $\rightarrow_P = \Rightarrow_{\mathcal{R}}$  if  $P$  is an elementary program  $\mathcal{R}$
- (2)  $\rightarrow_{\langle P_1; P_2 \rangle} = \rightarrow_{P_1} \circ \rightarrow_{P_2}$
- (3)  $\rightarrow_{P\downarrow} = \{ \langle G, H \rangle \mid G \rightarrow_P^* H \text{ and } H \text{ irreducible} \}$

## 8.2 Example: programs

### Generating a constant graph

$$\text{Const}_C = \langle \text{Delete}\downarrow; \text{Add}_C \rangle$$

where

### Reversing edges

$$\text{Converse} = \langle \text{Reverse}\downarrow; \text{Relabel}\downarrow \rangle$$

where

## 8.3 Completeness

A partial function  $f: \mathcal{A}_{C_1} \rightarrow \mathcal{A}_{C_2}$  on abstract graphs is **computable** if there exists a computable partial function  $f': \Sigma_1^* \rightarrow \Sigma_2^*$  on strings such that for every  $w \in \text{Exp}$ ,

$$f(\text{gra}(w)) = \text{gra}(f'(w))$$

and  $\text{gra}(w) \notin \text{Dom}(f)$  implies  $w \notin \text{Dom}(f')$ .

$$\begin{array}{lcl}
\text{Delete :} & \left\{ \begin{array}{l} \begin{array}{c} \textcircled{A}^1 \\ a \downarrow \\ \textcircled{B}^2 \end{array} \Rightarrow \begin{array}{c} \textcircled{A}^1 \\ \textcircled{B}^2 \end{array} \\ \textcircled{A} \Rightarrow \emptyset \end{array} & \begin{array}{l} A, B \in C_V, a \in C_E \\ A \in C_V \end{array} \\
\text{Add}_C : & \emptyset \Rightarrow C & 
\end{array}$$

Abbildung 1: The rules of  $\text{Const}_C$ 

$$\begin{array}{lcl}
\text{Reverse :} & \begin{array}{c} \textcircled{A}^1 \\ a \downarrow \\ \textcircled{B}^2 \end{array} \Rightarrow \begin{array}{c} \textcircled{A}^1 \\ a \uparrow \\ \textcircled{B}^2 \end{array} & A, B \in C_V, a \in C_E \\
\text{Relabel :} & \begin{array}{c} \textcircled{A}^1 \\ a \downarrow \\ \textcircled{B}^2 \end{array} \Rightarrow \begin{array}{c} \textcircled{A}^1 \\ a \downarrow \\ \textcircled{B}^2 \end{array} & A, B \in C_V, a \in C_E
\end{array}$$

Abbildung 2: The rules of  $\text{Converse}$ 

$$\begin{array}{ccc}
\mathcal{A}_{C_1} & \xrightarrow{f} & \mathcal{A}_{C_2} \\
\text{gra} \uparrow & & \uparrow \text{gra} \\
\Sigma_1^* & \xrightarrow{f'} & \Sigma_2^*
\end{array}$$

## 8.4 Graph Expressions

Given: alphabets  $C_V$ ,  $C_E$  of node and edge labels

Set  $\text{Exp}$  of graph expressions and graph  $w^\square$  for graph expression  $w$ :

- (1)  $\lambda \in \text{Exp}$  and  $\lambda^\square = \emptyset$
- (2) For  $A \in C_V$ :

$\#A1\# \in \text{Exp}$  and  $\#A1\#^\square$  contains a single node 1 with label  $A$

- (3) For  $v\#w \in \text{Exp}$  and  $A \in C_V$ :  
 $v\#A1^n\#w \in \text{Exp}$  with  $n = |V_{v\#w^\square}| + 1$  and  $v\#A1^n\#w^\square$  is obtained from  $v\#w^\square$  by adding node  $n$  with label  $A$
- (4) For  $v\#w \in \text{Exp}$  and  $F \in C_E$  where  $v\#w$  contains  $A1^m\#$  and  $B1^n\#$ :  
 $v\#A2^mF2^nB\#w \in \text{Exp}$  and  $v\#A2^mF2^nB\#w^\square$  is obtained from  $v\#w^\square$  by adding an edge with label  $F$ , source  $m$  and target  $n$

### 8.5 Representing abstract graphs

- $\mathcal{A}_C$ : set of all abstract graphs over  $C = \langle C_V, C_E \rangle$
- $\Sigma = C_V \cup C_E \cup \{1, 2, \#\}$

Partial function  $\text{gra}: \Sigma^* \rightarrow \mathcal{A}_C$ :

$$\text{gra}(w) = \begin{cases} [w^\square] & \text{if } w \text{ is a graph expression} \\ \text{undefined} & \text{otherwise.} \end{cases}$$

**Theorem 31. (Computational completeness)** For every computable partial function  $f$  there exists a program that computes  $f$ .

#### Minimality

A function  $f: \mathcal{A}_C \rightarrow \mathcal{A}_C$  is **cyclic** if there are some  $G$  and  $n \geq 2$  such that  $f(G) \neq G$  and  $f^n(G) = G$ .

**Lemma 32. (Cyclic function)** No cyclic function is computable by a program of the form  $P\downarrow$ .

#### Theorem 33. (Minimality)

- (1) The set of programs without sequential composition is computationally incomplete.
- (2) The set of programs without iteration is computationally incomplete.

## Conclusion

- Computationally complete core language for rule-based graph transformation:
  - nondeterministic one-step application of a set of rules
  - sequential composition
  - iteration in form of **apply as long as possible**
- Minimality: neither sequential composition nor iteration can be omitted
- Simple formal semantics
- Graph problems can be solved at a high level of abstraction

## Future work

- Enriching the language for more programming comfort:
  - rules with application conditions
  - more control constructs (e.g. more general conditional statement, repeat/while-loop)
  - procedures
  - (graph) types
- **Implementation**
- Using the core constructs over other rule-based frameworks (e.g. string and term rewriting)

## A Kategorientheorie

Einheitliche Behandlung gemeinsamer Konzepte aus unterschiedlichen mathematischen Theorien

- möglichst geringe Voraussetzungen
- gemeinsame Eigenschaften müssen nicht in jeder Theorie neu bewiesen werden

Typische Beispiele

- Mengen mit Funktionen, partiellen Funktionen oder Relationen
- topologische Räume mit stetigen Funktionen
- Monoide mit Monoid-Homomorphismen
- Gruppen mit Gruppen-Homomorphismen
- Ringe mit Ring-Homomorphismen
- endliche Automaten mit Automaten-Homomorphismen
- Graphen mit Graphmorphismen
- Hypergraphen mit Hypergraphmorphismen
- Stellen-Transitions-Netze mit Netz-Morphismen
- relationale Strukturen mit Struktur-Morphismen
- Spezifikationen mit Spezifikations-Morphismen

⋮

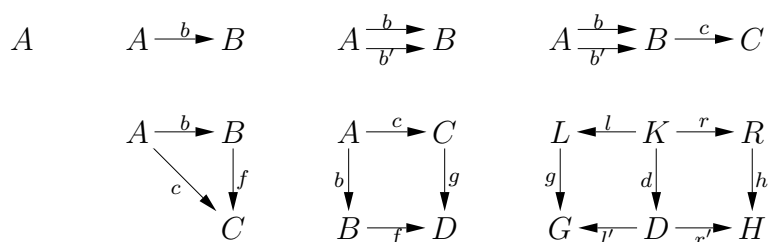
**Definition 24. (Kategorie)** Eine **Kategorie**  $K$  besteht aus einer Klasse  $\text{Obj}$  von **Objekten**, einer Menge  $\text{Mor}(A, B)$  von **Morphismen** für jedes Paar  $A, B \in \text{Obj}$  und einer **Komposition**  $\circ: \text{Mor}(A, B) \times \text{Mor}(B, C) \rightarrow \text{Mor}(A, C)$  für jedes Tripel  $A, B, C \in \text{Obj}$  derart, daß gilt:

- **Assoziativität:**  $h \circ (g \circ f) = (h \circ g) \circ f$  für alle Objekte  $A, B, C, D \in \text{Obj}$  und alle Morphismen  $f \in \text{Mor}(A, B)$ ,  $g \in \text{Mor}(B, C)$ ,  $h \in \text{Mor}(C, D)$ .

- **Identität:** Zu jedem Objekt  $A \in \text{Obj}$  existiert ein Morphismus  $\text{id}_A \in \text{Mor}(A, A)$ , die **Identität** auf  $A$  derart, daß gilt:  $f \circ \text{id}_A = f$  und  $\text{id}_A \circ g = g$  für alle Objekte  $B, C \in \text{Obj}$  und alle Morphismen  $f \in \text{Mor}(A, B)$ ,  $g \in \text{Mor}(C, A)$ .

**Schreibweisen:**  $f: A \rightarrow B$  oder  $A \xrightarrow{f} B$  für  $f \in \text{Mor}(A, B)$   
 $A \xrightarrow{f} B \xrightarrow{g} C$  für  $A \xrightarrow{g \circ f} C$

**Definition 25. (Diagramme)** Ein **Diagramm** (in einer Kategorie) ist ein gerichteter Graph, dessen Knoten mit Objekten und dessen Kanten mit Morphismen markiert sind, derart, daß gilt: Ist  $e$  eine Kante mit Quelle  $v$ , Ziel  $v'$  und Markierung  $f \in \text{Mor}(A, B)$ , so ist  $v$  mit  $A$  und  $v'$  mit  $B$  markiert.



Jeder Weg von einem mit  $A$  markierten Knoten zu einem mit  $B$  markierten Knoten repräsentiert einen Morphismus in  $\text{Mor}(A, B)$ . Ein Diagramm heißt **kommutativ**, wenn je zwei Wege mit dem gleichen Start- und Zielknoten den gleichen Morphismus repräsentieren.



$f: A \rightarrow B$  heißt **Isomorphismus**, wenn ein Morphismus  $g: B \rightarrow A$  existiert, so daß  $g \circ f = \text{id}_A$  und  $f \circ g = \text{id}_B$  gilt. In diesem Fall heißen die Objekte  $A$  und  $B$  **isomorph**, in Zeichen  $A \cong B$ .

$$\begin{array}{ccc}
 A & \xrightarrow{f} & B \xrightarrow{g} A \\
 \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} \\
 & \text{id}_A & \text{id}_B
 \end{array}
 \qquad
 \begin{array}{ccc}
 B & \xrightarrow{g} & A \xrightarrow{f} B \\
 \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} \\
 & \text{id}_B & \text{id}_A
 \end{array}$$

**Definition 26. (Pushouts)**  $B \xrightarrow{f} D \xleftarrow{g} C$  heißt **Pushout** (kurz PO) zu  $B \xleftarrow{b} A \xrightarrow{c} C$ , wenn gilt:

1. **Kommutativität:**  $f \circ b = g \circ c$
2. **Universelle Eigenschaft:** Für alle Objekte  $D'$  und alle Morphismen  $f': B \rightarrow D'$ ,  $g': C \rightarrow D'$  mit  $f' \circ b = g' \circ c$  existiert genau ein Morphismus  $u: D \rightarrow D'$  mit  $u \circ f = f'$  und  $u \circ g = g'$ .

$$\begin{array}{ccccc}
 A & \xrightarrow{c} & C & & \\
 \downarrow b & = & \downarrow g & \searrow g' & \\
 B & \xrightarrow{f} & D & \xrightarrow{u} & D' \\
 & \searrow f' & & & \\
 & & & & D'
 \end{array}$$

**Lemma 34. (Eindeutigkeit)** Pushout-Objekte sind bis auf Isomorphie eindeutig.

**Beweis.** Seien  $ABCD$  und  $ABCD'$  Pushouts. Da  $ABCD$  ein Pushout und  $ABCD'$  kommutativ ist, existiert genau ein Morphismus  $u: D \rightarrow D'$  mit  $u \circ f = f'$  und  $u \circ g = g'$ . Da  $ABCD'$  ein Pushout und  $ABCD$  kommutativ ist, existiert genau ein Morphismus  $u': D' \rightarrow D$  mit  $u' \circ f' = f$  und  $u' \circ g' = g$ . Andererseits existiert genau ein Morphismus  $u'': D \rightarrow D$  mit  $u'' \circ f = f$  und  $u'' \circ g = g$ , da  $ABCD$  ein Pushout und kommutativ ist. Also gilt  $u' \circ u = u'' = \text{id}_D$ . Analog folgt  $u \circ u' = \text{id}_{D'}$ . Also ist  $u: D \rightarrow D'$  ein Isomorphismus und  $D \cong D'$ .  $\square$

**Definition 27. (Pushout-Komplemente)**  $A \rightarrow^c C \rightarrow^g D$  heißt **Pushout-Komplement** (kurz PO-Komplement) zu  $A \rightarrow^b B \rightarrow^f D$ , wenn  $B \rightarrow^f D \leftarrow^g C$  Pushout zu  $B \leftarrow^b A \rightarrow^c C$  ist.

$$\begin{array}{ccc} A & \xrightarrow{b} & B \\ \downarrow c & \text{(PO)} & \downarrow f \\ C & \xrightarrow{g} & D \end{array}$$

**Definition 28. (Pullbacks)**  $B \leftarrow^f D \rightarrow^g C$  heißt **Pullback** (kurz PB) zu  $B \rightarrow^b A \leftarrow^c C$ , wenn gilt

1. **Kommutativität:**  $b \circ f = c \circ g$
2. **Universelle Eigenschaft:** Für alle Objekte  $D'$  und alle Morphismen  $f': D' \rightarrow B$ ,  $g': D' \rightarrow C$  mit  $b \circ f' = c \circ g'$  existiert genau ein Morphismus  $u: D' \rightarrow D$  mit  $f \circ u = f'$  und  $g \circ u = g'$ .

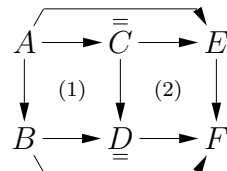
$$\begin{array}{ccc} D' & & \\ \downarrow f' & \searrow g' & \\ \downarrow f & \xrightarrow{g} & C \\ \downarrow f & \searrow g & \downarrow c \\ B & \xrightarrow{b} & A \end{array}$$

**Lemma 35. (Eindeutigkeit)** Pullback-Objekte sind bis auf Isomorphie eindeutig.

**Beweis.** Analog zum Beweis der Eindeutigkeit von Pushouts.  $\square$

1. Sind die Diagramme (1) und (2) Pushout-Diagramme, so ist auch das Diagramm (3) ein Pushout-Diagramm.
2. Sind die Diagramme (1)+(2) und (1) Pushout-Diagramme, so ist auch das Diagramm (2) ein Pushout-Diagramm.





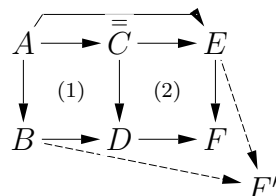
**Bemerkung.** Es läßt sich **nicht allgemein** zeigen, daß gilt:

3. Sind die Diagramme (1)+(2) und (2) Pushout-Diagramme, so ist auch das Diagramm (1) ein Pushout-Diagramm.

(Dies gilt nur in speziellen Kategorien unter speziellen Voraussetzungen!)

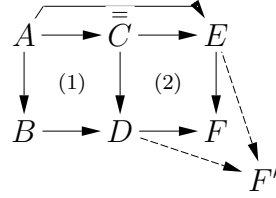
**Beweis.**

1. Seien (1) und (2) PO's. Zu zeigen: (1)+(2) PO.



Sei  $F'$  ein beliebiges Objekt und  $B \rightarrow F'$ ,  $E \rightarrow F'$  Morphismen derart, daß  $A \rightarrow B \rightarrow F' = A \rightarrow E \rightarrow F'$ . Definiere  $C \rightarrow F' = C \rightarrow E \rightarrow F'$ . Da (1) PO und  $A \rightarrow B \rightarrow F' = A \rightarrow E \rightarrow F' = A \rightarrow C \rightarrow E \rightarrow F' = A \rightarrow C \rightarrow F'$  ist, existiert genau ein Morphismus  $D \rightarrow F'$  mit  $B \rightarrow D \rightarrow F' = B \rightarrow F'$  und  $C \rightarrow D \rightarrow F' = C \rightarrow F'$ . Da (2) PO und  $C \rightarrow D \rightarrow F' = C \rightarrow F' = C \rightarrow E \rightarrow F'$  ist, existiert genau ein Morphismus  $F \rightarrow F'$  mit  $D \rightarrow F \rightarrow F' = D \rightarrow F'$  und  $E \rightarrow F \rightarrow F' = E \rightarrow F'$ . Damit existiert genau ein Morphismus  $F \rightarrow F'$  mit  $B \rightarrow F \rightarrow F' = B \rightarrow D \rightarrow F \rightarrow F' = B \rightarrow D \rightarrow F' = B \rightarrow F'$  und  $E \rightarrow F \rightarrow F' = E \rightarrow F'$ . Also ist (1)+(2) PO.

2. Seien (1)+(2), (1) PO. Zu zeigen: (2) PO.



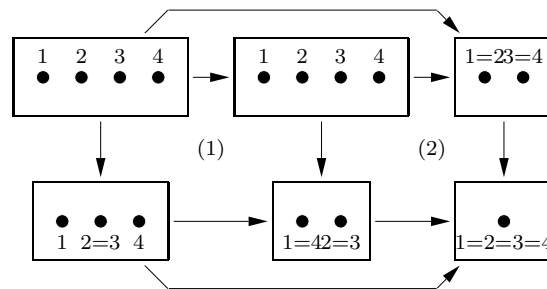
Sei  $F'$  ein beliebiges Objekt und  $D \rightarrow F'$ ,  $E \rightarrow F'$  Morphismen derart, daß  $C \rightarrow D \rightarrow F' = C \rightarrow E \rightarrow F'$ . Definiere  $B \rightarrow F' = B \rightarrow D \rightarrow F'$  und  $C \rightarrow F' = C \rightarrow E \rightarrow F'$ . Da (1)+(2) PO und  $A \rightarrow B \rightarrow F' = A \rightarrow B \rightarrow D \rightarrow F' = A \rightarrow C \rightarrow D \rightarrow F' = A \rightarrow C \rightarrow E \rightarrow F' = A \rightarrow E \rightarrow F'$  ist, existiert genau ein Morphismus  $F \rightarrow F'$  mit  $B \rightarrow F \rightarrow F' = B \rightarrow F'$  und  $E \rightarrow F \rightarrow F' = E \rightarrow F'$ . Es bleibt zu zeigen, daß  $D \rightarrow F \rightarrow F' = D \rightarrow F'$  gilt.

Für den zusammengesetzten Morphismus  $D \rightarrow F \rightarrow F'$  gilt:  $B \rightarrow D \rightarrow F \rightarrow F' = B \rightarrow F \rightarrow F' = B \rightarrow F'$  und  $C \rightarrow D \rightarrow F \rightarrow F' = C \rightarrow E \rightarrow F \rightarrow F' = C \rightarrow E \rightarrow F' = C \rightarrow F'$ .

Für den Morphismus  $D \rightarrow F'$  gilt:  $B \rightarrow D \rightarrow F' = B \rightarrow F'$  und  $C \rightarrow D \rightarrow F' = C \rightarrow E \rightarrow F' = C \rightarrow F'$ . Da (1) PO und  $A \rightarrow B \rightarrow F' = A \rightarrow B \rightarrow D \rightarrow F' = A \rightarrow C \rightarrow D \rightarrow F' = A \rightarrow C \rightarrow E \rightarrow F' = A \rightarrow C \rightarrow F'$  ist, existiert genau ein Morphismus  $D \rightarrow F'$  mit dieser Eigenschaft. Also gilt  $D \rightarrow F \rightarrow F' = D \rightarrow F'$ . Damit ist (2) PO.

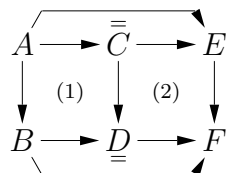
3. Seien (1)+(2), (2) PO. Zu zeigen: i.a. gilt nicht: (1) PO.

In der Kategorie GRAPHS:



□

**Lemma 37. (Kompositions- und Dekompositionslemma für Pullbacks)** Sind die Kompositionen (1) und (2) Dekompositionen, so ist auch das zusammengesetzte Diagramm (1)+(2) ein Pullback-Diagramm.



2. Sind die Diagramme (1)+(2) und (2) Pullback-Diagramme, so ist auch das Diagramm (1) ein Pullback-Diagramm.

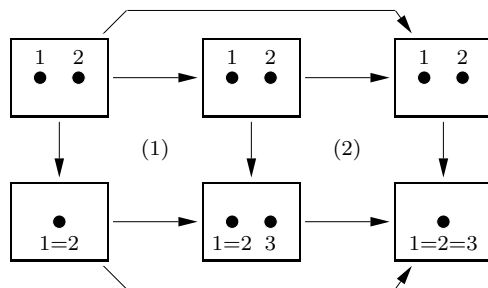
**Beweis.** Ähnlich zum Beweis des Kompositions- und Dekompositionslemma für Pushouts.  $\square$

**Bemerkung.** Es läßt sich **nicht allgemein** zeigen, daß gilt:

3. Sind die Diagramme (1)+(2) und (1) Pullback-Diagramme, so ist auch das Diagramm (2) ein Pullback-Diagramm.

(Dies gilt nur in speziellen Kategorien unter speziellen Voraussetzungen!)

Seien (1)+(2), (1) PB. Zu zeigen: i.a. gilt nicht: (2) PB.  
In der Kategorie GRAPHS:



**Die Kategorie SETS**

Objekte: Mengen  
Morphismen: Abbildungen  
Komposition: Komposition von Abbildungen

**Die Kategorie GRAPHS**

**Lemma 38. (Kategorie GRAPHS)** Die Graphen über  $C = (C_V, C_E)$  und Graphmorphismen bilden zusammen mit der Komposition eine Kategorie GRAPHS.

**Beweis.** Übung □

**Lemma 39. (Isomorphismus in GRAPHS)** Ein Graphmorphismus  $f: A \rightarrow B$  ist genau dann ein Isomorphismus in der Kategorie GRAPHS, wenn die Abbildungen  $f_V, f_E$  bijektiv sind.

**Beweis.** Übung □

**Lemma 40. (Konstruktion von Pushouts in GRAPHS)** Seien  $b: A \rightarrow B, c: A \rightarrow C$  Graphmorphismen. Dann liefert die folgende Verklebungskonstruktion ein Pushout.

- Sei  $D = (V_D, E_D, s_D, t_D, l_D, m_D)$  mit

$$V_D = (V_B + V_C) / \approx \text{ und } E_D = (E_B + E_C) / \approx$$

wobei  $\approx$  ist die von  $\sim$  erzeugte Äquivalenzrelation ist und  $\sim$  durch  $b(x) \sim c(x)$  für  $x \in A$  definiert ist,

$s_D([e]) = \text{if } e \in E_B \text{ then } [s_B(e)] \text{ else } [s_C(e)], t_D([e])$  analog,

$l_D([v]) = \text{if } v \in V_B \text{ then } l_B(v) \text{ else } l_C(v), m_D([e])$  analog.

- Seien  $f: B \rightarrow D, g: C \rightarrow D$  die dazugehörigen natürlichen Graphmorphismen ( $f(x) = [x]$  für  $x \in B$  und  $g(x) = [x]$  für  $x \in C$ ).

$$\begin{array}{ccc}
A & \xrightarrow{b} & B \\
c \downarrow & (1) & \downarrow f \\
C & \xrightarrow{g} & D
\end{array}$$

**Lemma 41. (spezielle Konstruktion von Pushouts in GRAPHS)** Seien  $b: A \rightarrow B$ ,  $c: A \rightarrow C$  Graphmorphisimen und  $b: A \rightarrow B$  injektiv. Dann liefert die folgende Verklebungskonstruktion ein Pushout.

1. Sei  $D = (V_D, E_D, s_D, t_D, l_D, m_D)$  mit  $D = C + (B - b(A))$  für die Menge der Knoten bzw. Kanten, wobei  $s_D$ ,  $t_D$ , und  $m_D$  für alle  $e \in E_D$  und  $l_D$  für alle  $v \in V_D$  wie folgt definiert sind:

$$\begin{aligned}
s_D(e) &= \text{if } e \in E_C \text{ then } s_C(e) \text{ else} \\
&\quad \text{if } s_B(e) \in V_B - b_V(V_A) \text{ then } s_B(e) \\
&\quad \text{else } c_V(v) \text{ wobei } s_B(e) = c_V(v) \text{ für } v \in V_A
\end{aligned}$$

$t_D(e)$  analog

$$l_D(v) = \text{if } v \in V_B \text{ then } l_B(v) \text{ else } l_C(v)$$

$m_D(e)$  analog

2.  $f: B \rightarrow D$  ist die Inklusion von  $B$  in  $D$ .
3.  $g: C \rightarrow D$  ist für alle  $x \in C$  durch  $g(y) = \text{if } y \in C - c(A) \text{ then } y \text{ else } b(x)$ , wobei  $y = c(x)$  für  $x \in A$ .

**Beweis.** (siehe Ehrig 79, Beweis von Lemma 2.8 in 9.6). □

**Lemma 42. (Konstruktion von Pullbacks in GRAPHS)**

$$b: A \rightarrow B$$

$$c: A \rightarrow C$$

Seien  $b: B \rightarrow A$ ,  $c: C \rightarrow A$  Graphmorphisimen. Dann liefert die folgende Konstruktion ein Pullback.

- $V_D = \{\langle x, y \rangle \in V_B \times V_C \mid b_V(x) = c_V(y)\}$
- $E_D = \{\langle x, y \rangle \in E_B \times E_C \mid b_E(x) = c_E(y)\}$

- $s_D(\langle x, y \rangle) = \langle s_B(x), s_D(y) \rangle$  für  $\langle x, y \rangle \in E_D$
- $t_D(\langle x, y \rangle) = \langle t_B(x), t_D(y) \rangle$  für  $\langle x, y \rangle \in E_D$
- $l_D(\langle x, y \rangle) = l_B(x) \quad [= l_C(y)]$  für  $\langle x, y \rangle \in V_D$
- $m_D(\langle x, y \rangle) = m_B(x) \quad [= m_C(y)]$  für  $(x, y) \in E_D$

Seien  $f: D \rightarrow B$ ,  $g: D \rightarrow C$  die Graphmorphismen mit  $f((x, y)) = x$  und  $g((x, y)) = y$  für  $(x, y) \in D$ .

**Lemma 43. (Existenz von Pushouts und Pullbacks)**

1. Die Kategorie GRAPHS besitzt Pushouts, d.h. für jedes Paar  $B: A \rightarrow B$ ,  $c: A \rightarrow C$  von Graphmorphismen existiert ein Pushout  $B \rightarrow D \leftarrow C$  zu  $B \leftarrow A \rightarrow C$ .
2. Die Kategorie GRAPHS besitzt Pullbacks, d.h. für jedes Paar  $B \rightarrow A$ ,  $C \rightarrow A$  von Graphmorphismen existiert ein Pullback  $B \leftarrow D \rightarrow C$  zu  $B \rightarrow A \leftarrow C$ .

**Beweis.** (folgt direkt aus den Konstruktion von Pushouts und Pullbacks). □

Im folgenden sei (1) das folgende kommutative Diagramm:

$$\begin{array}{ccc} A & \xrightarrow{b} & B \\ c \downarrow & (1) & \downarrow f \\ C & \xrightarrow{g} & D \end{array}$$

**Lemma 44. (Injektivitätslemma für Pushouts und Pullbacks)** Sei (1) ein kommutatives Diagramm.

1. Ist (1) ein Pushout-Diagramm und  $A \rightarrow B$  injektiv, so ist auch  $C \rightarrow D$  injektiv.
2. Ist (1) ein Pullback-Diagramm und  $C \rightarrow D$  injektiv, so ist auch  $A \rightarrow B$  injektiv.

**Beweis.** (folgt direkt aus der Konstruktion von Pushouts und Pullbacks) □

$$\begin{array}{ccc}
 A & \xrightarrow{b} & B \\
 \downarrow c & (1) & \downarrow f \\
 C & \xrightarrow{g} & D
 \end{array}$$

### A.1 Eigenschaften von Graphmorphismen

- $f: B \rightarrow D$  heißt **injektiv bis auf  $b: A \rightarrow B$** , wenn für alle  $x, x' \in B$  mit  $x \neq x'$  und  $f(x) = f(x')$  gilt:  $x, x' \in b(A)$ .  $f: B \rightarrow D$  heißt **injektiv bis auf  $g: C \rightarrow D$** , wenn für alle  $x, x' \in B$  mit  $x \neq x'$  und  $f(x) = f(x')$  gilt:  $f(x) = f(x') \in g(C)$ .
- $(f: B \rightarrow D, g: C \rightarrow D)$  heißt **gemeinsam surjektiv**, wenn  $f(B) \cup g(C) = D$ .
- $(f: B \rightarrow D, g: C \rightarrow D)$  erfüllt die **Kettenbedingung**, wenn für alle  $y \in B$  und alle  $z \in C$  mit  $f(y) = g(z)$ ,  $x_1, \dots, x_{2n+1} \in A$  existieren, so daß gilt:  $b(x_1) = y$ ,  $c(x_{2n+1}) = z$  und  $c(x_{2i-1}) = b(x_{2i})$  und  $b(x_{2i}) = c(x_{2i+1})$  für  $i = 1, \dots, n$ .  
 $(f: B \rightarrow D, g: C \rightarrow D)$  erfüllt die **reduzierte Kettenbedingung**, wenn für alle  $y \in B$  und alle  $z \in D$  mit  $f(y) = g(z)$ , ein  $x \in A$  existiert, so daß  $b(x) = y$  und  $c(x) = z$ .
- $(b: A \rightarrow B, c: A \rightarrow C)$  heißt **monomorph**, wenn für alle  $x, x' \in A$  mit  $b(x) = b(x')$  und  $c(x) = c(x')$  gilt:  $x = x'$ .

**Theorem 45. (Pushout-Charakterisierung, Ehrig, Kreowski 79, Theorem 1.2)** Sei (1) ein kommutatives Diagramm. Dann ist (1) genau dann ein Pushout-Diagramm, wenn

- (a)  $f$  ist injektiv bis auf  $g$ ,
- (b)  $g$  ist injektiv bis auf  $f$ ,
- (b)  $(f, g)$  sind gemeinsam surjektiv,
- (d)  $(f, g)$  erfüllt die Kettenbedingung in (1).

Die Bedingungen (a) und (b) können durch (a') und (b') ersetzt werden:

- (a')  $f$  ist injektiv bis auf  $b$ ,
- (b')  $g$  ist injektiv bis auf  $c$ .

Falls  $b$  oder  $c$  injektiv ist, kann (d) durch (d') ersetzt werden:

(d')  $(f, g)$  erfüllt die reduzierte Kettenbedingung in (1).

**Theorem 46. (Pullback-Charakterisierung, Ehrig, Kreowski 79, Theorem 1.7)** Sei (1) ein kommutatives Diagramm. Dann ist (1) genau dann ein Pullback-Diagramm, wenn

- (a)  $(b, c)$  ist monomorph,
- (b)  $(f, g)$  erfüllt die reduzierte Kettenbedingung in (1).

**Lemma 47. (Pullback-Pushout-Lemma, Ehrig, Kreowski 79, Korollar 1.1)** Ist (1) ein Pullback-Diagramm, sind  $(B \rightarrow D, C \rightarrow D)$  gemeinsam surjektiv und  $B \rightarrow D$  und  $C \rightarrow D$  injektiv, so ist (1) auch ein Pushout-Diagramm.

**Beweis.** (mit der Pullback- und Pushout-Charakterisierung) Da nach Voraussetzung  $B \rightarrow D$  und  $C \rightarrow D$  injektiv,  $(B \rightarrow D, C \rightarrow D)$  gemeinsam surjektiv und für Pullbacks die reduzierte Kettenbedingung erfüllt ist, ist ein (1) Pushout-Diagramm.  $\square$

**Lemma 48. (Pushout-Pullback-Lemma, Ehrig, Kreowski 79, Korollar 1.9)** Ist (1) ein Pushout-Diagramm und  $A \rightarrow B$  injektiv, so ist (1) auch ein Pullback-Diagramm.

**Beweis.** (mit der Pushout- und Pullback-Charakterisierung) Da  $A \rightarrow B$  injektiv ist, ist  $(A \rightarrow B, A \rightarrow C)$  monomorph. Da (1) Pushout-Diagramm ist und  $A \rightarrow B$  injektiv ist, erfüllt  $(B \rightarrow D, C \rightarrow D)$  die reduzierte Kettenbedingung. Mit der Pullback-Charakterisierung folgt, daß (1) Pullback-Diagramm ist.  $\square$

## A.2 Verklebungsbedingung

$f: B \rightarrow D$  erfüllt die **Verklebungsbedingung** bzgl.  $b: A \rightarrow B$ , wenn  $\text{Glue}(f) \subseteq b(A)$ . Dabei ist

$$\begin{aligned} \text{Glue}(f) &= \text{Dang}(f) \cup \text{Ident}(f) \text{ mit } \text{Dang}(f) = \{x \in V_B \mid \exists a \in \\ E_D - f_E(E_B): f_V(x) &= s_G(a) \vee f_V(x) = t_G(a)\}, \\ \text{Ident}(f) &= \{x \in B \mid \exists x' \in B \mid x \neq x' \wedge f(x) = f(x')\} \end{aligned}$$



**Lemma 49. (Konstruktion von Pushout-Komplementen)** Erfülle  $f: B \rightarrow D$  die **Verklebungsbedingung** bzgl.  $b: A \rightarrow B$  und sei  $b: A \rightarrow B$  injektiv. Sei  $D$  der Graph mit ... und seien  $d: K \rightarrow D$ ,  $l^*: D \rightarrow G$  die Graphmorphismen mit  $d(x) = gl(x)$  für  $x \in K$  und  $l^*(x) = x$  für  $x \in D$ . Dann ist  $K \rightarrow D \rightarrow G$  ein Pushout-Komplement zu  $K \rightarrow L \rightarrow G$ .

**Lemma 50. (Existenz und Eindeutigkeit von Pushout-Komplementen)**

1. Seien  $b: A \rightarrow B$ ,  $f: B \rightarrow D$  Graphmorphismen. Dann existiert genau dann ein Pushout-Komplement  $A \rightarrow C \rightarrow D$  zu  $A \rightarrow B \rightarrow D$ , wenn  $f: B \rightarrow D$  die Verklebungsbedingung bzgl.  $b: A \rightarrow B$  erfüllt.
2. Das Pushout-Komplement von  $A \rightarrow B \rightarrow D$  ist bis auf Isomorphie eindeutig, wenn  $b: A \rightarrow B$  injektiv ist.

**Lemma 51. (Spezielles Dekompositionslemma)** Sei ein kommutatives

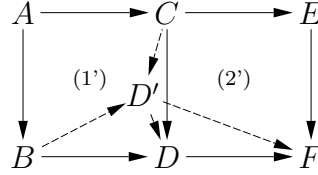
$$\begin{array}{ccccc}
 A & \xrightarrow{\quad} & C & \xrightarrow{\quad} & E \\
 \downarrow & (1) & \downarrow & (2) & \downarrow \\
 B & \xrightarrow{\quad} & D & \xrightarrow{\quad} & F
 \end{array}$$

Diagramm.

1. Sind (1)+(2) und (2) Pushout-Diagramme und ist  $C \rightarrow E$  injektiv, so ist auch (1) ein Pushout-Diagramm.
2. Ist (1)+(2) ein Pullback-Diagramm, (1) ein Pushout-Diagramm und  $E \rightarrow F$  injektiv, so ist (2) ein Pullback-Diagramm.

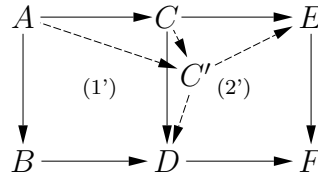
**Beweis.**

1. Seien (1)+(2) und (2) PO's und  $C \hookrightarrow E$  injektiv. Zu zeigen: (1) PO. Konstruiere das Pushout  $B \rightrightarrows D' \leftarrow C$  von  $B \leftarrow A \rightarrow C$ . Bezeichne das entstehende Diagramm mit (1'). Da (1') PO und (1) kommutativ ist, existiert genau ein Morphismus  $D' \rightarrow D$  mit  $B \rightarrow D' \rightarrow D = B \rightarrow$



$D$  und  $C \rightarrow D' \rightarrow D = C \rightarrow D$ . Definiere  $D' \rightarrow F = D' \rightarrow D \rightarrow F$ . Dann ist das entstehende Diagramm (2') kommutativ:  $C \rightarrow D' \rightarrow F = C \rightarrow D' \rightarrow D \rightarrow F = C \rightarrow D \rightarrow F = C \rightarrow E \rightarrow F$ . Wegen  $B \rightarrow F = B \rightarrow D \rightarrow F = B \rightarrow D' \rightarrow D \rightarrow F = B \rightarrow D' \rightarrow F$  ist  $(1)+(2)=(1')+(2')$  PO. Da weiterhin (1') PO ist, folgt aus dem Dekompositionslemma für PO's, daß (2') PO ist. Da (2) und (2') PO's sind, sind  $C \rightarrow D \rightarrow F$  und  $C \rightarrow D' \rightarrow F$  PO-Komplemente von  $C \rightarrow E \rightarrow F$ . Da  $C \rightarrow E$  injektiv ist, existiert bis auf Isomorphie genau ein PO-Komplement. Also muß  $D' \rightarrow D$  ein Isomorphismus und (1) PO sein.

2. Sei (1)+(2) PB, (1) PO und  $E \rightarrow F$  injektiv. Zu zeigen: (1) PB. Kon-

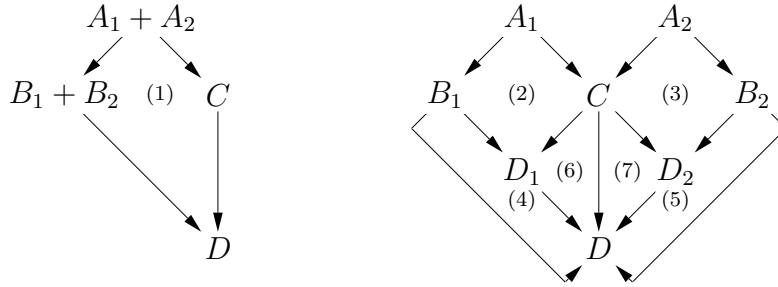


struiere das Pullback  $D \leftarrow C' \rightarrow E$  von  $D \rightarrow F \leftarrow E$ . Bezeichne das entstehende Diagramm mit (2'). Dann existiert genau ein Morphismus  $C \rightarrow C'$  mit  $C \rightarrow C' \rightarrow D = C \rightarrow D$  und  $C \rightarrow C' \rightarrow E = C \rightarrow E$ . Ziel ist, zu zeigen, daß  $C \rightarrow C'$  ein Isomorphismus ist. Definiere  $A \rightarrow C' = A \rightarrow C \rightarrow C'$ . Dann ist das entstehende Diagramm (1') kommutativ:  $A \rightarrow B \rightarrow D = A \rightarrow C \rightarrow D = A \rightarrow C \rightarrow C' \rightarrow D = A \rightarrow C' \rightarrow D$ . Wegen  $A \rightarrow E = A \rightarrow C \rightarrow E = A \rightarrow C \rightarrow C' \rightarrow E = A \rightarrow C' \rightarrow E$  ist  $(1)+(2)=(1')+(2')$  PB. Da weiterhin (2') PB ist, folgt aus dem Dekompositionslemma für PB's, daß (1') PB ist. Mit dem Injektivitätslemma für PB's folgt aus der Injektivität von  $E \rightarrow F$  die Injektivität von  $A \rightarrow B$  und  $C' \rightarrow D$ . Mit dem Injektivitätslemma für PO's folgt aus der Injektivität von  $A \rightarrow B$  die Injektivität von  $C \rightarrow D$ . Da  $C \rightarrow D = C \rightarrow C' \rightarrow D$  und  $C \rightarrow D$  injektiv ist, ist  $C \rightarrow C'$  injektiv.

$C \rightarrow C'$  ist auch surjektiv: Da (1) PO ist, ist  $(B \rightarrow D, C \rightarrow D)$  gemeinsam surjektiv, d.h.  $D = (B \rightarrow D)(B) \cup (C \rightarrow D)(C)$ . Sei nun  $x' \in C'$  und  $y$  das Bild von  $x'$  in  $D$ . Ist  $y$  in  $(B \rightarrow D)(B)$ , so existiert wegen der PB-Eigenschaft von (1') ein  $a \in A$  mit  $(A \rightarrow C)(a) = x'$  und ein  $x = (A \rightarrow C)(a) \in C$  mit  $(C \rightarrow C')(x) = x'$ . Ist  $y$  in  $(C \rightarrow D)(C)$ , so existiert ein  $x \in C$  mit  $(C' \rightarrow D)(C \rightarrow C')(x) = y = (C' \rightarrow D)(x')$  und, da  $C' \rightarrow D$  injektiv ist, ist  $(C \rightarrow C')(x) = x'$ .

□

**Lemma 52. (Schmetterlingslemma, Kreowski 77, Lemma 1.6)** Das Diagramm (1) ist genau dann ein Pushout-Diagramm, wenn es eine Zerlegung in die kommutativen Diagramme (2) bis (7) gibt derart, daß die Diagramme (2), (3) und (8) := (6)  $\cup$  (7) Pushout-Diagramme sind. (Hierbei



seien  $A_i \rightarrow B_i$ ,  $A_i \rightarrow C$  und  $B_i \rightarrow D$  ( $i = 1, 2$ ) Graphmorphismen und  $A_1 + A_2 \rightarrow B_1 + B_2$ ,  $A_1 + A_2 \rightarrow C$  und  $B_1 + B_2 \rightarrow D$  die eindeutig induzierten Graphmorphismen.)

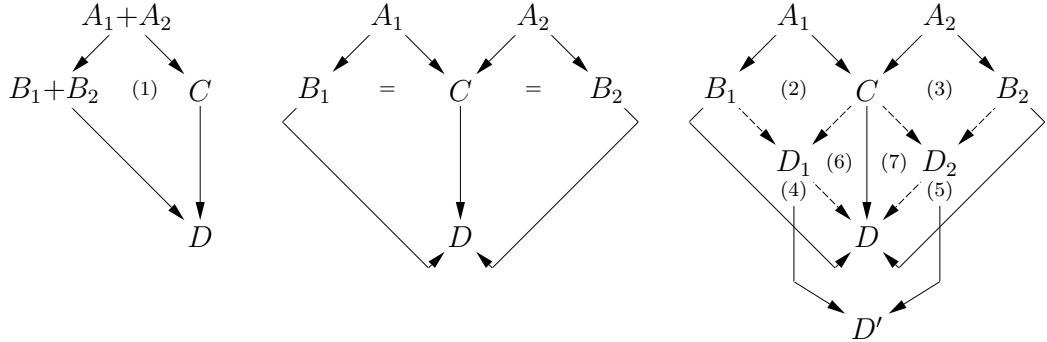
**Beweis.** “ $\Rightarrow$ ” Sei (1) ein PO-Diagramm.

Konstruiere  $D_i$  als PO-Objekt von  $B_i \leftarrow A_i \rightarrow C$  ( $i = 1, 2$ ).

Da  $A_i \rightarrow B_i \rightarrow D = A_i \rightarrow B_i \rightarrow B_1 + B_2 \rightarrow D = A_i \rightarrow B_1 + B_2 \rightarrow D = A_i \rightarrow A_1 + A_2 \rightarrow B_1 + B_2 \rightarrow D = A_i \rightarrow A_1 + A_2 \rightarrow C \rightarrow D = A_i \rightarrow C \rightarrow D$  ( $i = 1, 2$ ) ist, existieren Graphmorphismen  $D_i \rightarrow D$  derart, daß  $B_i \rightarrow D_i \rightarrow D = B_i \rightarrow D$  und  $C \rightarrow D_i \rightarrow D = C \rightarrow D$  ( $i = 1, 2$ ) gilt. Damit sind die entstehenden Diagramme (4), (5), (6), (7) und (8) = (6)  $\cup$  (7) kommutativ.

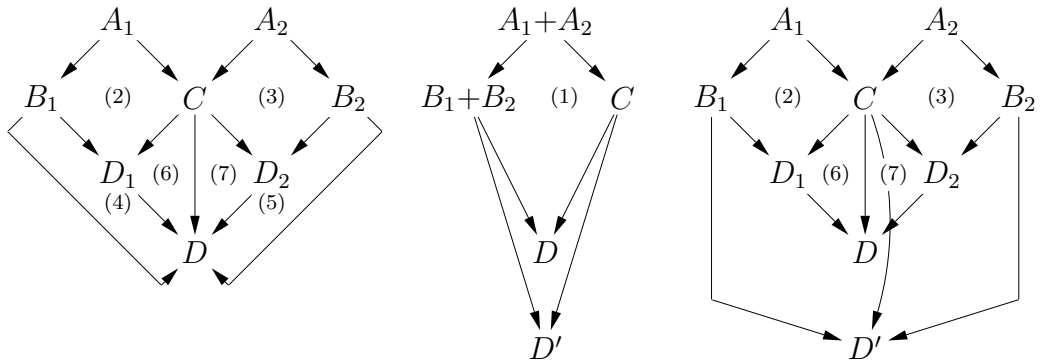
Es bleibt zu zeigen, daß (8) = (6)  $\cup$  (7) Pushout-Diagramm ist.

Sei nun  $D'$  ein beliebiger Graph und seien  $D_i \rightarrow D'$  Graphmorphismen derart, daß  $C \rightarrow D_1 \rightarrow D' = C \rightarrow D_2 \rightarrow D'$ . Definiere  $B_i \rightarrow D' = B_i \rightarrow D_i \rightarrow D'$



( $i = 1, 2$ ). Sei  $B_1 + B_2 \rightarrow D'$  der induzierte Graphmorphismus und  $C \rightarrow D' = C \rightarrow D_1 \rightarrow D'$ . Dann kommutiert das entstehende Diagramm. Da (1) PO-Diagramm ist, existiert eindeutig ein Graphmorphismus  $D \rightarrow D'$  derart, daß  $B_1 + B_2 \rightarrow D \rightarrow D' = B_1 + B_2 \rightarrow D' = C \rightarrow D \rightarrow D' = C \rightarrow D'$  gilt. Damit ist  $B_i \rightarrow D_i \rightarrow D \rightarrow D' = B_i \rightarrow D \rightarrow D' = B_i \rightarrow B_1 + B_2 \rightarrow D \rightarrow D' = B_i \rightarrow B_1 + B_2 \rightarrow D' = B_i \rightarrow D' = B_i \rightarrow D_i \rightarrow D' = C \rightarrow D_i \rightarrow D \rightarrow D' = C \rightarrow D \rightarrow D'$ . Da nach Konstruktion (2) und (3) PO-Diagramme sind, sind  $(B_i \rightarrow D_i, C \rightarrow D_i)$  ( $i = 1, 2$ ) gemeinsam surjektiv. Folglich gilt  $D_i \rightarrow D \rightarrow D' = D_i \rightarrow D'$  ( $i = 1, 2$ ).

“ $\Leftarrow$ ” Es existiere eine Zerlegung in die kommutativen Diagramme (2) bis (7) derart, daß (2), (3) und (8) := (6)  $\cup$  (7) PO-Diagramme sind. Zu zeigen: (1) ist ein PO-Diagramm. (1) ist kommutativ, da (2)–(7) kommutativ sind. Sei



nun  $D'$  ein beliebiger Graph und seien  $B_1 + B_2 \rightarrow D'$ ,  $C \rightarrow D'$  Graphmorphis-  
men derart, daß  $A_1 + A_2 \rightarrow B_1 + B_2 \rightarrow D' = A_1 + A_2 \rightarrow C \rightarrow D'$ . Definiere  
 $B_i \rightarrow D' = B_i \rightarrow B_1 + B_2 \rightarrow D'$  ( $i = 1, 2$ ). Dann ist  $A_i \rightarrow B_i \rightarrow D' =$   
 $A_i \rightarrow C \rightarrow D'$ .

Da (2) und (3) PO-Diagramme sind, existieren eindeutig Graphmorphismen  $D_i \rightarrow D'$  derart, daß  $B_i \rightarrow D_i \rightarrow D' = B_i \rightarrow D'$  und  $C \rightarrow D_i \rightarrow D' = C \rightarrow D'$  ( $i = 1, 2$ ). Da (8) ein PO-Diagramm ist, existiert eindeutig ein Graphmorphismus  $D \rightarrow D'$  mit  $D_i \rightarrow D \rightarrow D' = D_i \rightarrow D'$  ( $i = 1, 2$ ). Folglich ist  $B_i \rightarrow B_1+B_2 \rightarrow D \rightarrow D' = B_i \rightarrow D \rightarrow D' = B_i \rightarrow D' = B_i \rightarrow B_1+B_2 \rightarrow D'$  ( $i = 1, 2$ ). Da die Graphmorphismen  $(B_1 \rightarrow B_1+B_2, B_2 \rightarrow B_1+B_2)$  gemeinsam surjektiv sind, gilt  $B_1+B_2 \rightarrow D \rightarrow D' = B_1+B_2 \rightarrow D'$ . Da (6) und (7) kommutativ sind, ist  $C \rightarrow D \rightarrow D' = C \rightarrow D_i \rightarrow D \rightarrow D' = C \rightarrow D_i \rightarrow D' = C \rightarrow D'$ .  $\square$

### A.3 Zusammenfassung

- in jeder Kategorie:
  - Eindeutigkeit von PO's und PB's
  - Kompositionslemmata für PO's und PB's
  - Deompositionslemmata für PO's und PB's
- in der Kategorie GRAPHS:
  - Konstruktion von PO's und PB's ( $\rightarrow$  Existenz von PO's und PB's)
  - Charakterisierung von PO's und PB's
  - Injektivitätslemmata für PO's und PB's
  - PO-PB-Lemma und PB-PO-Lemma
  - spezielle Deompositionslemmata für PO's und PB's
  - Schmetterlingslemma

## Literatur

- [BC87] Michel Bauderon, Bruno Courcelle. Graph expressions and graph rewriting. *Mathematical Systems Theory* 20, 83–127, 1987.
- [Cou90] Bruno Courcelle. The monadic second-order logic of graphs I: Recognizable sets of finite graphs. *Information and Computation* 85, 12–75, 1990.
- [Ehr79] Hartmut Ehrig. Introduction to the algebraic theory of graph grammars. In *Graph-Grammars and Their Application to Computer Science and Biology*, volume 73 of *Lecture Notes in Computer Science*, 1–69. Springer-Verlag, 1979.
- [EPS73] Hartmut Ehrig, Michael Pfender, Hans Jürgen Schneider. Graph grammars: An algebraic approach. In *Proc. 14th Annual IEEE Symposium on Switching and Automata Theory*, 167–180, Iowa City, 1973.
- [ER91] Joost Engelfriet, Grzegorz Rozenberg. Graph grammars based on node rewriting: An introduction to NLC graph grammars. In *Graph Grammars and Their Application to Computer Science*, volume 532 of *Lecture Notes in Computer Science*, 12–23. Springer-Verlag, 1991.
- [Göt88] Herbert Göttler. Graphgrammatiken in der Softwaretechnik, volume 178 of *Informatik-Fachberichte*. Springer-Verlag, Berlin, 1988.
- [Hab92] Annegret Habel. Hyperedge Replacement: Grammars and Languages, volume 643 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1992.
- [HK87] Annegret Habel, Hans-Jörg Kreowski. May we introduce to you: Hyperedge replacement. In *Graph Grammars and Their Application to Computer Science*, volume 291 of *Lecture Notes in Computer Science*, 15–26. Springer-Verlag, 1987.
- [JR80] Dirk Janssens, Grzegorz Rozenberg. On the structure of node-label-controlled graph languages. *Information Sciences* 20, 191–216, 1980.
- [Nag73] Manfred Nagl. Eine Präzisierung des Pfaltz/Rosenfeldschen Produktionsbegriffs bei mehrdimensionalen Grammatiken. *Arbeitsbericht* 6a, 3, 56–71, Institut für Mathematische Maschinen und Datenverarbeitung, Erlangen, 1973.
- [Nag79] Manfred Nagl. *Graph-Grammatiken: Theorie, Anwendungen, Implementierungen*. Vieweg, Braunschweig, 1979.

- [Plu93] Detlef Plump. Hypergraph rewriting: Critical pairs and undecidability of confluence. In *Term Graph Rewriting: Theory and Practice*, 201–213. John Wiley, New York, 1993.
- [Roz87] Grzegorz Rozenberg. An introduction to the NLC way of rewriting graphs. In *Graph Grammars and Their Application to Computer Science*, volume 291 of *Lecture Notes in Computer Science*, 55–66. Springer-Verlag, 1987.

# Index

- track-Funktion, 67
- Ableitungen
  - direkte, 22
  - Eigenschaften von, 36
  - konfluente, 62
  - Parallelableitung, 58
  - parallele Unabhängigkeit, 54
  - sequentielle Unabhängigkeit, 51
  - Termination von, 64
- Beispiel
  - Bibliothekssystem, 28
    - Graphersetzungssystem BIB, 29
    - Graphgrammatik BIB, 35
  - Flußdiagramme, 14, 37, 40
    - Graphgrammatik WSF, 24
  - zusammenhängende Graphen, 37
    - Graphgrammatik CON, 24
- Chomsky-Grammatik, 46
- Einbettungssatz, 39
- Einschränkungssatz, 40
- Graph, 9
- Graphausdruck, 74
- Graphersetzungsregel, 15
- Graphersetzungssystem, 23
- Graphgrammatik, 23
- Graphisomorphie, 10
- Graphmorphismus, 11
- Graphprogramme, 73
  - Minimalität, 75
  - Vollständigkeit, 73
- Identifikationsbedingung, 18
- Kategorie, 77
  - GRAPHS, 84
- Konfluenz, 62
  - lokale Konfluenz, 63
- Kontaktbedingung, 17
- Kritisches Paar, 66
- Markierung, 9
- Markierungsalphabet, 9
- Noethersche Induktion, 65
- Normalform, 62
- PCP, Postsches Korrespondenzproblem, 48
- Pullback, 80
- Pullback-Charakterisierung, 88
- Pushout, 79
- Pushout-Charakterisierung, 87
- Pushout-Komplement, 80
- Quelle, 9
- Teilgraph, 12
- Termination, 64
- Turingmaschine, 44
- Verklebungsbedingung, 88
- Ziel, 9