

# Hypergraph-Grammatiken mit Dschungel-Eigenschaft und ihr Verhältnis zu L-attribuierten Grammatiken

Dem Fachbereich Mathematik  
der Universität Hannover  
zur Erlangung des Grades  
Doktor der Naturwissenschaften

Dr. rer. nat.  
genehmigte Dissertation

von

Dipl.-Inform. Henning Ehlermann  
geboren am 2. September 1964 in Hünzingen

1997

Referent: Prof. Dr. J. Duske  
Korreferent: Prof. Dr. R. Parchmann

Tag der Promotion: 31.10.1997

## Zusammenfassung der wesentlichen Ergebnisse

In der vorliegenden Arbeit werden kontextfreie Hypergraph-Grammatiken betrachtet, die die Eigenschaft haben, daß alle Satzformen Dschungel sind — Dschungel wiederum eignen sich gut zur Festlegung von Termen.

Es wird gezeigt, daß solche Hypergraph-Grammatiken bestimmte weitere Eigenschaften z. B. bezüglich ihrer Produktionen haben. Diese leichter festzustellenden Eigenschaften werden genutzt, um konstruktiv nachzuweisen, daß die durch solche Hypergraph-Grammatiken festgelegte Familie von Termsprachen identisch ist mit jener, die durch L-attributierte Grammatiken bzw. IO-Makrogrammatiken festgelegt werden – dies wurde bereits von J. Engelfriet und L. Heyker (Context-free hypergraph grammars have the same term-generating power as attribute grammars. *Acta Informatica* **29**, 161-210 (1992)) vermutet.

Darüber hinaus werden verschiedene Formen von linearen Hypergraph-Grammatiken (mit der Eigenschaft, daß alle Satzformen Dschungel sind) beschrieben und verglichen.

Es wird über den Iterationssatz von A. Habel ein iteratives Verfahren zur Erzeugung von Termen für IO-Makrosprachen angegeben.

Außerdem werden zwei Normalformen für solche Hypergraph-Grammatiken vorgestellt — die in Anlehnung an kontextfreie Grammatiken Chomsky- und Greibach-Normalform genannt werden.

Schlagnworte: kontextfreie Hypergraph-Grammatik, Dschungel-Eigenschaft, L-attributierte Grammatik

# Abstract

We are investigating context-free hypergraph grammars with the property that all sentential forms are jungles. Jungles are useful because they can be transformed into terms in a quite obvious way.

There are additional properties of such hypergraph grammars given. These properties are used to show constructively that the family of term languages defined by hypergraph grammars is equal to the family of term languages defined by L-attributed grammars or by IO macro grammars respectively. This result has been conjectured by J. Engelfriet and L. Heyker in [EnHe92].

Different forms of linear hypergraph grammars are given and discussed.

Based on the pumping lemma given by A. Habel, an algorithm to generate terms of IO macro grammars in an iterative way is presented.

Two normal forms for these hypergraph grammars are proposed. Due to the similarity to context-free grammars they are called Chomsky and Greibach normal form.

keywords: context-free hypergraph grammar, jungle property, L-attributed grammar

# Inhaltsverzeichnis

0.1	Einleitung . . . . .	7
<b>1</b>	<b>Grundlegende Definitionen</b>	<b>9</b>
1.1	Alphabet und Term . . . . .	9
1.2	Kontextfreie Hypergraph-Grammatik . . . . .	10
1.3	L-attributierte Grammatik . . . . .	19
1.4	IO-Makrogrammatik . . . . .	23
1.5	Ordnung muß sein . . . . .	26
<b>2</b>	<b>Dschungel und Grammatikeigenschaften</b>	<b>33</b>
2.1	Reiner Dschungel und Grammatikeigenschaften . . . . .	36
<b>3</b>	<b>Der Iterationssatz im Dschungel</b>	<b>45</b>
3.1	Der Iterationssatz . . . . .	45
<b>4</b>	<b>Lineare Hypergraph-Grammatiken</b>	<b>63</b>
4.1	Formen der Linearität . . . . .	63
4.2	Sprachfamilien . . . . .	77
<b>5</b>	<b>Kontextfreie Hypergraph-Grammatiken und L-attributierte Grammatiken</b>	<b>81</b>
5.1	Simulation von L-attributierten Grammatiken durch kontextfreie Hypergraph-Grammatiken mit Dschungel-Eigenschaft . . . . .	81
5.1.1	Konstruktion . . . . .	84

5.1.2	Vorbereitungen . . . . .	92
5.1.3	Ableitungsteilbäume . . . . .	93
5.1.4	Nachweis der Gleichheit . . . . .	97
5.2	Simulation von Hypergraph-Grammatiken mit Dschungel-Eigenschaft durch L-attributierte Grammatiken . . . . .	100
5.2.1	Konstruktion . . . . .	100
5.2.2	Vorbereitungen . . . . .	102
5.2.3	Nachweis der Gleichheit . . . . .	103
<b>6</b>	<b>IO-Makrogrammatiken und kontextfreie Hypergraph-Grammatiken</b>	<b>109</b>
6.1	Simulation von kontextfreien Hypergraph-Grammatiken mit Dschungel-Eigenschaft durch IO-Makrogrammatiken . . . . .	109
6.1.1	Konstruktion . . . . .	110
6.1.2	Zwischenüberlegungen . . . . .	114
6.1.3	Weitere Vorbereitungen . . . . .	116
6.1.4	Nachweis der Gleichheit . . . . .	119
6.2	Simulation von IO-Makrogrammatiken durch kontextfreie Hypergraph-Grammatiken mit Dschungel-Eigenschaft . . . . .	122
6.2.1	Konstruktion . . . . .	122
6.2.2	Beweisidee . . . . .	123
<b>7</b>	<b>Weitere Betrachtungen</b>	<b>125</b>
7.1	Der Iterationssatz und IO-Makrogrammatiken . . . . .	125
7.2	Normalformen . . . . .	136
7.2.1	Chomsky-Normalform . . . . .	136
7.2.2	Greibach-Normalform . . . . .	142
7.3	Ausblick . . . . .	150

# Abbildungsverzeichnis

0.1	Kapitelübersicht . . . . .	8
1.1	Hyperkante vom Rang 3 . . . . .	10
1.2	Skizze eines baumähnlichen Hypergraphen . . . . .	16
1.3	Skizze eines baumähnlichen Hypergraphen . . . . .	17
1.4	Produktion $p_1$ für Fibonacci . . . . .	19
1.5	Produktion $p_2$ für Fibonacci . . . . .	20
1.6	Produktion $p_3$ für Fibonacci . . . . .	20
1.7	Beispiel einer Ableitung . . . . .	21
2.1	Produktion $p_1$ . . . . .	39
2.2	Produktion $p_2$ . . . . .	40
2.3	Ableitung in $G$ . . . . .	40
2.4	Produktion $p'_1$ . . . . .	43
2.5	Produktion $p'_2$ . . . . .	43
2.6	Ableitung in $G'$ . . . . .	44
3.1	Skizze eines Ableitungsbaums . . . . .	46
3.2	Ableitung von „ $S \Rightarrow^k uAz \Rightarrow^L uvAyz$ “ . . . . .	48
3.3	Ableitung von „ $uvAyz \Rightarrow^m uvxyz$ “ . . . . .	49
3.4	Ableitung von „ $S \Rightarrow^k uAz \Rightarrow^m uxz$ “ . . . . .	50
3.5	Andere Darstellung der Hypergraphen $H_1$ , $H_2$ und $H_3$ aus Abbildung 3.2 und 3.3 . . . . .	51

3.6	„Skizze“ des Iterationssatzes . . . . .	52
3.7	Produktionen von $G$ . . . . .	55
3.8	„ $h(f^3(1), g^3(1))$ “ . . . . .	56
3.9	Zerlegung von „ $h(f(1), g(1))$ “ . . . . .	57
3.10	Beginn der Ableitung . . . . .	58
3.11	Andere Darstellung der Zerlegung aus Abbildung 3.9 . . . . .	59
4.1	Produktionen aus $P_u$ . . . . .	64
4.2	Produktionen aus $P_o$ . . . . .	65
4.3	Produktionen aus $P_l$ . . . . .	66
4.4	Produktionen aus $P_{li}$ . . . . .	68
4.5	Zerlegung von „ $f^{36}(1)$ “ . . . . .	69
4.6	Produktionen aus $P_k$ . . . . .	70
4.7	Ableitung in $G_k$ . . . . .	71
4.8	Zerlegung von „ $f^2(g^2(1)) \in L(G_l)$ “ . . . . .	73
4.9	„Andere“ Zerlegung von „ $f^2(g^2(1)) \in L(G_l)$ “ . . . . .	73
4.10	Produktionen aus $P$ . . . . .	74
4.11	Zerlegung von „ $f^4(1) \in L(G)$ “ . . . . .	75
4.12	„Andere“ Zerlegung von „ $f^4(1)$ “ . . . . .	75
4.13	„Andere“ Zerlegung von „ $f^4(1)$ “ . . . . .	76
4.14	„Andere“ Zerlegung von „ $f^4(1)$ “ . . . . .	76
4.15	Ersetzungsregel 1 zur Umwandlung von untenlinear nach obenlinear	79
4.16	Ersetzungsregel 2 zur Umwandlung von untenlinear nach obenlinear	79
4.17	Ersetzungsregel 3 zur Umwandlung von untenlinear nach obenlinear	80
4.18	Ersetzungsregel 4 zur Umwandlung von untenlinear nach obenlinear	80
5.1	Eine L-attributierte Produktion und die dazu konstruierte rechte Seite einer Hypergraph Produktion . . . . .	82
5.2	Schritt 1 . . . . .	88

5.3	Schritt 2 . . . . .	88
5.4	Schritt 3 . . . . .	89
5.5	Schritt 4 . . . . .	90
5.6	Schritt 5 . . . . .	91
5.7	Zu $A \rightarrow BC$ konstruierte Produktion . . . . .	93
6.1	Hypergraph-Produktionen . . . . .	110
6.2	Hypergraph und $Term_M$ bzw. $Term_E$ . . . . .	113
6.3	Ausschnitt aus Ableitungsteilbaum mit $p^i$ als Wurzel . . . . .	115
6.4	Ausschnitt aus Ableitungsteilbaum mit $\tilde{p}_0^i$ als Wurzel . . . . .	116
7.1	„Allgemeine“ Zerlegung in <i>first</i> , <i>link</i> und <i>last</i> . . . . .	126
7.2	Produktionen aus $G$ . . . . .	129
7.3	Produktionen aus $G$ nach Beseitigung der $\epsilon$ -Produktion . . . . .	130
7.4	Produktionen zu $G_2$ . . . . .	131
7.5	Ableitungsbaum für $1^3(c1^3)^7$ . . . . .	132
7.6	<i>first</i> von „ $1^3(c1^3)^7$ “ . . . . .	133
7.7	<i>link</i> von „ $1^3(c1^3)^7$ “ . . . . .	134
7.8	<i>last</i> von „ $1^3(c1^3)^7$ “ . . . . .	135
7.9	Problematische Produktion . . . . .	137
7.10	Noch problematischere Produktionen . . . . .	137
7.11	Singuläre Produktion (Nummerierung der unteren externen Knoten beachten !) . . . . .	138
7.12	Zu der Produktion aus Abbildung 7.11 konstruierte Produktionen . . . . .	138
7.13	Einführung neuer terminaler Produktionen . . . . .	139
7.14	Beseitigung terminaler Hyperkanten in den übrigen Produktionen . . . . .	140
7.15	Kontraktion . . . . .	141
7.16	„Muster“ einer rechten Seite in Greibach-Normalform . . . . .	143
7.17	Alle Produktionen mit $A$ auf der linken Seite . . . . .	144

7.18 ... werden ersetzt durch diese und ...	145
7.19 ... diese Produktionen	146
7.20 Hier ist $D_i$ ganz unten auf der rechten Seite	147
7.21 Alle Produktionen mit $D_i$ auf der linken Seite	148
7.22 Ersatz für die Produktion aus Abbildung 7.20	148

## 0.1 Einleitung

In den letzten Jahren haben Hypergraph-Grammatiken zunehmend an Bedeutung gewonnen. Dies beruht u. a. auf den erweiterten Möglichkeiten, die sie im Vergleich zu „herkömmlichen“ Grammatiken bieten. Die neuen Aspekte eröffnen ein weites, erst lückenhaft erforschtes Feld. Von großem Interesse sind dabei die Auswirkungen von Einschränkungen, um möglichst sinnvolle Beschränkungen zu finden.

Dschungel sind spezielle Hypergraphen, die sich insbesondere dadurch auszeichnen, daß man ihnen einfach und eindeutig Terme zuordnen kann.

Die vorliegende Arbeit konzentriert sich daher auf Hypergraph-Grammatiken, die Dschungel erzeugen und deren Satzformen ebenfalls durchweg Dschungel sind.

Das Hauptaugenmerk liegt dabei auf Zusammenhänge zwischen einerseits Hypergraph-Grammatiken und andererseits L-attribuierten Grammatiken und IO-Makrogrammatiken. Hierzu wird zunächst (in Kapitel 2) festgestellt, daß diese Hypergraph-Grammatiken bestimmte weitere Eigenschaften z. B. bezüglich ihrer Produktionen besitzen müssen. Dann werden (in Kapitel 5 und 6) Konstruktionsverfahren angegeben, die zu Hypergraph-Grammatiken mit solchen Eigenschaften L-attribuierte Grammatiken bzw. IO-Makrogrammatiken erzeugen und umgekehrt. Anschließend wird gezeigt, daß die vorgegebene und die konstruierte Grammatik jeweils die gleiche Termsprache festlegen. Daraus folgt die Identität der Familien der Termsprachen. Da Duske e. a. (vergleiche [DuPaSeSp]) bereits festgestellt haben, daß zwischen L-attribuierten Grammatiken und IO-Makrogrammatiken diese Beziehung vorliegt, ist eine dieser beiden Identitäten aus der anderen zu folgern, aber die Angabe des direkten Weges ist hilfreich für weitere Untersuchungen.

Da die erwähnten Konstruktionsverfahren mit den dazugehörigen Betrachtungen nicht ganz leicht verständlich sind, wird bereits vor diesen das Kapitel 4 über lineare Hypergraph-Grammatiken eingeschoben. Diese Anordnung soll Lesern, die sich noch nicht intensiv mit Hypergraph-Grammatiken auseinandergesetzt haben, einen besseren Einstieg zu geben. Dort werden drei Formen von linearen Hypergraph-Grammatiken (deren Satzformen Dschungel sind) und die Beziehungen zwischen den dadurch jeweils festgelegten Sprachfamilien angegeben.

Hierfür wird u. a. der Iterationssatz von A. Habel benötigt. Dieser wird in Kapitel 3 vorgestellt.

Da für IO-Makrogrammatiken bisher kein Iterationssatz bekannt ist, ist untersucht worden, welche Aspekte sich durch den Iterationssatz von A. Habel und den in Kapitel 6 angegebenen Konstruktionsverfahren ergeben. Die Ergebnisse werden in Kapitel 7 vorgestellt. Dort werden auch zwei Normalformen für Hypergraph-Grammatiken (deren Satzformen Dschungel sind) angegeben.

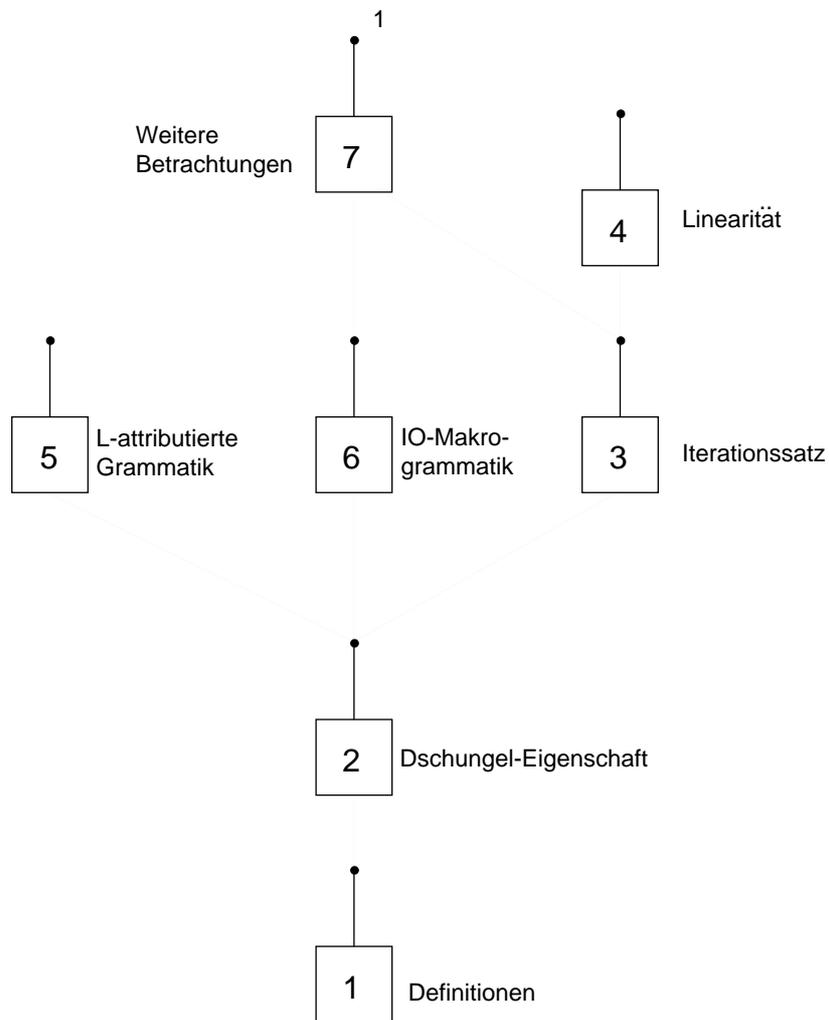


Abbildung 0.1: Kapitelübersicht

In der Abbildung 0.1 sind die Abhängigkeiten zwischen den Kapiteln graphisch dargestellt. Ein Kapitel hängt von den Definitionen und Ergebnissen jener darunter liegenden Kapitel ab, zu denen eine direkte oder indirekte („abwärtsgerichtete“) Verbindung besteht. Das Bild kann auch als Dschungel interpretiert werden (zur Markierung des externen Knotens wurde deswegen die „1“ am obersten Knoten angefügt).

Meinen Dank aussprechen möchte ich dem Institut für Informatik, das mir die Möglichkeit zur Promotion gab, und meinem Doktorvater Prof. Dr. rer. nat. J. Duske für die Betreuung dieser Arbeit.

# Kapitel 1

## Grundlegende Definitionen

In diesem Kapitel werden die grundlegenden Begriffe und Definitionen vorgestellt. Dabei wird eine gewisse Vertrautheit mit der Theorie der formalen Sprachen vorausgesetzt. Außerdem werden in Form von Bemerkungen und Folgerungen einige einfache Zusammenhänge aufgezeigt.

Da zumindest ein Teil der einzuführenden Begriffe bereits in der bestehenden Literatur verwendet wurde, scheint es nicht sinnvoll, hierfür ganz neue Schreib- und Ausdrucksweisen zu definieren. Stattdessen lehnen sich z.B. die Definitionen, die die Hypergraphen betreffen, häufig an der von J. Engelfriet (siehe z. B. [EnHe92]) und anderen verwendeten Nomenklatur an.

### 1.1 Alphabet und Term

**Definition 1.1 (Alphabet)** *Ein Alphabet  $V$  ist eine endliche, nichtleere Menge, deren Elemente auch als Buchstaben bezeichnet werden.*

*Eine endliche Folge von Buchstaben aus  $V$  heißt Wort über  $V$ .*

*Ist  $w = a_1 \dots a_n$  ein Wort über  $V$ , dann heißt  $|w| := n$  Länge von  $w$ .*

*Das Wort mit der Länge 0 heißt leeres Wort und wird mit  $\epsilon$  bezeichnet.*

*Die Menge aller Worte über  $V$  wird mit  $V^*$  bezeichnet.*

$V^+ := V^* \setminus \epsilon$ .

**Definition 1.2 (Marken-Alphabet)** *Ein Alphabet  $\Sigma$  mit einer Abbildung  $rank_\Sigma : \Sigma \rightarrow \mathbb{N}_0$  heißt Marken-Alphabet.*

*$rank_\Sigma(e)$ ,  $e \in \Sigma$ , heißt Rang von  $e$ .*

**Bemerkung:**

Ist aus dem Zusammenhang klar, auf welches Alphabet  $\Sigma$  sich  $rank_\Sigma$  bezieht, wird statt  $rank_\Sigma$  auch nur  $rank$  geschrieben.

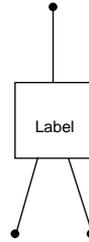


Abbildung 1.1: Hyperkante vom Rang 3

**Definition 1.3 (Term)** Sei  $\mathcal{F}$  ein Marken-Alphabet.

Seien die Sonderzeichen „(“, „),“ und „{, }“ nicht in  $\mathcal{F}$  enthalten.

Ein Term über  $\mathcal{F}$  ist ein Wort über  $\mathcal{F} \cup \{(\} \cup \{,\} \cup \{\}\}$ , wenn folgendes erfüllt ist:

- a) Falls  $a \in \mathcal{F}$  und  $\text{rank}(a) = 0$  ist, dann ist  $a()$  ein Term und (als Kurzform)  $a$  ein Term.
- b) Falls  $F \in \mathcal{F}$  ist und  $t_1, \dots, t_{\text{rank}(F)}$  Terme sind, dann ist auch  $F(t_1, \dots, t_{\text{rank}(F)})$  ein Term.
- c) Ein Wort über  $\mathcal{F} \cup \{(\} \cup \{,\} \cup \{\}\}$  ist ein Term, wenn es durch endlich viele Anwendungen der Regeln a) und b) konstruiert werden kann.

## 1.2 Kontextfreie Hypergraph-Grammatik

Da Hypergraphen und im speziellen Dschungel eine große Rolle in dieser Arbeit spielen werden, ist der folgende Abschnitt der Definition dieser Begriffe und einiger mit ihnen in engem Zusammenhang stehender Begriffe vorbehalten.

Ein Hypergraph unterscheidet sich von einem Graphen im wesentlichen dadurch, daß eine Hyperkante im Gegensatz zu einer „normalen“ Kante nicht mit genau zwei Knoten verbunden sein muß. Stattdessen kann z.B. eine Hyperkante vom Rang 3 mit drei Knoten verbunden sein. Graphisch wird eine solche Hyperkante durch ein Kästchen mit drei „Tentakeln“ dargestellt (siehe Abbildung 1.1).

Es besteht die Möglichkeit, den Hyperkanten Marken – genauer Buchstaben aus einem Marken-Alphabet – zuzuordnen:

**Definition 1.4 (Hypergraph)** Sei  $\Sigma$  ein Marken-Alphabet.

Ein Hypergraph über  $\Sigma$  ist ein 5-Tupel  $H = (V, E, \text{nod}, \text{lab}, \text{ext})$  mit

$V$  ist eine endliche Menge, die Menge der Knoten (nodes),  
 $E$  ist eine endliche Menge, die Menge der Hyperkanten (hyperedges),  
 $nod : E \rightarrow V^*$  ist die Inzidenz-Funktion,  
 $lab : E \rightarrow \Sigma$  ist die Markierungs-Funktion (labeling-) und  
 $ext \in V^*$ .

Für jede Hyperkante  $e \in E$  muß gelten:  $rank_{\Sigma}(lab(e)) = |nod(e)|$ .

Sei  $ext = v_1 \dots v_n$ ,  $n \geq 0$ . Dann heißt die Menge  $\{v_i \mid i \in [1 : n]\}$  die Menge der externen Knoten von  $H$ , die anderen interne Knoten.

$v_n$  wird auch designierter Knoten von  $H$  genannt.

Es werden die folgenden Sprechweisen vereinbart:

**Definition 1.5 (Bezeichnungen)** Sei  $H = (V, E, nod, lab, ext)$  ein Hypergraph.  
 $m = |ext|$  ist der Rang von  $H$ , geschrieben  $rank(H)$ , und  $H$  wird  $m$ -Hypergraph genannt.

Die Menge aller Hypergraphen über  $\Sigma$  wird mit  $HGR(\Sigma)$  bezeichnet.

Sei  $nod(e) = v_1 \dots v_r$ , dann wird  $r$  Rang von  $e$  genannt ( $rank(e) = r$ ) und  $e$  wird  $r$ -Hyperkante genannt.

Der Knoten  $v_i$  wird mit  $nod(e, i)$  bezeichnet.

Sei  $ext = w_1 \dots w_m$ , dann ist  $ext(j) = w_j$ .

Eine Hyperkante  $e$  wird als Eingangs-Kante des Knotens  $v$  bezeichnet, wenn  $nod(e, rank(e)) = v$  gilt.

Der Knoten  $v$  wird als one-incoming bezeichnet, wenn er genau eine Eingangs-Kante in  $H$  besitzt.

Wenn  $v = nod(e, i)$  gilt, so wird dies anschaulich damit beschrieben, daß die Hyperkante  $e$  ein Tentakel zum Knoten  $v$  besitzt. Dieses Tentakel wird mit  $i$  markiert.

Gilt  $nod(e, rank(e)) = v$ , so wird  $v$  auch Ausgangsknoten von  $e$  und das zugehörige Tentakel Ausgangstentakel genannt.

Jetzt sind wir in der Lage, eine kontextfreie Hypergraph-Grammatik zu definieren:

**Definition 1.6 (kontextfreie Hypergraph-Grammatik)**

Eine kontextfreie Hypergraph-Grammatik (context-free hypergraph grammar) ist ein 4-Tupel  $G = (\Sigma, \Delta, P, S)$  mit

$\Sigma$  ist ein Marken-Alphabet,

$\Delta \subset \Sigma$  ist das Alphabet der Terminalen,

$\Sigma - \Delta$  ist das Alphabet der Nichtterminalen,

$S \in \Sigma - \Delta$  das Startsymbol,

$P$  ist eine endliche Menge von Paaren  $(X, H)$ ,  $X \in \Sigma - \Delta$ ,  $H \in HGR(\Sigma)$ ,

$rank_{\Sigma}(X) = rank(H)$ , und heißt Menge der Produktionen von  $G$ .

Falls  $ext = v_1 \dots v_m$ ,  $m \geq 1$  ist, muß gelten:  $v_i = v_j \Rightarrow i = j$ ,  $i, j \in [1 : m]$ .

**Bemerkung:**

Die Forderung, daß die externen Knoten des Hypergraphen  $H$  paarweise disjunkt sind, bedeutet, daß die Grammatik gemäß Engelfriet ([EnHe92], Seite 371) *identification-free* ist.

**Definition 1.7** Sei  $G = (\Sigma, \Delta, P, S)$  eine kontextfreie Hypergraph-Grammatik und  $H = (V, E, \text{nod}, \text{lab}, \text{ext})$  ein Hypergraph über  $\Sigma$ . Dann ist

$|H| = |E|$  (Anzahl der Hyperkanten in  $H$ ) und

$|H|_N = |\{e \in E \mid \text{lab}(e) \in \Sigma - \Delta\}|$  (Anzahl der Hyperkanten in  $H$  mit nichtterminalem Label).

**Definition 1.8 (lhs(p), rhs(p))** Sei  $G = (\Sigma, \Delta, P, S)$  eine kontextfreie Hypergraph-Grammatik und  $p = (A, H) \in P$ . Dann ist

$\text{lhs}(p) = A$  (left-hand side) und

$\text{rhs}(p) = H$  (right-hand side).

Eine kontextfreie Hypergraph-Grammatik  $G$  soll eine Menge von Hypergraphen erzeugen. Diese Menge wird dann Sprache von  $G$  genannt. Bevor wir dies definieren können, brauchen wir aber den Begriff der Ableitung und hierzu wiederum das „replacement“:

**Definition 1.9 (replacement)** Seien  $H = (V_H, E_H, \text{nod}_H, \text{lab}_H, \text{ext}_H)$  und  $R = (V_R, E_R, \text{nod}_R, \text{lab}_R, \text{ext}_R)$  Hypergraphen,

$e$  eine Hyperkante von  $H$  ( $e \in E_H$ ),  $\text{rank}(e) = m$ ,  $m \geq 1$ ,

$\text{ext}_R = v_1 \dots v_m$  mit  $v_i = v_j \Rightarrow i = j$ ,  $i, j \in [1 : m]$  und

$\text{EXT}_R = \{v_i \in V_R \mid i \in [1 : m]\}$

Dann ist  $\text{repl}(e|R, H) = X$ ,  $X = (V_X, E_X, \text{nod}_X, \text{lab}_X, \text{ext}_X)$ , die Ersetzung (replacement) der Hyperkante  $e$  durch  $R$  in  $H$  mit:

- $V_X = V_H \cup V_R \setminus \text{EXT}_R$
- $E_X = E_R \cup E_H \setminus \{e\}$
- alle Hyperkanten behalten ihre Label.
- $\text{nod}_X$ :
  - Alle Hyperkanten aus  $H$  behalten ihre Verbindungen ( $\forall e'' \in E_H \setminus \{e\}$  :  $\text{nod}_X(e'', i) = \text{nod}_H(e'', i)$ )
  - Die Hyperkanten aus  $R$  behalten ihre Verbindungen zu internen Knoten von  $R$  ( $\forall e' \in E_R \forall v' \in V_R \setminus \text{EXT}_R : \text{nod}_X(e', i) = v' \Leftrightarrow \text{nod}_R(e', i) = v'$ )
  - $\text{nod}_X(e', i) = v$ ,  $e' \in E_R$ ,  $v \in V_H \Leftrightarrow \text{nod}_R(e', i) = v_j$ ,  $j \in [1 : m] \wedge \text{nod}_H(e, j) = v$

- $ext_X = ext_H$

Anstelle von  $repl(e|R, H)$  ist auch die Schreibweise  $H[e|R]$  zugelassen.

**Definition 1.10 (Ableitung)** Sei  $G = (\Sigma, \Delta, P, S)$ ,  $(X, H) = p \in P$ ,  $H_a, H_b$  Hypergraphen über  $\Sigma$  und  $e$  eine Kante aus  $H_a$  mit  $lab(e) = X$ .

Dann kann  $H_b$  in einem Schritt aus  $H_a$  mit  $p$  bzgl.  $e$  abgeleitet werden,  $H_a \Rightarrow_e^p H_b$ , wenn  $H_b = repl(e|H, H_a)$  gilt.

Es ist  $H_b$  in  $G$  direkt ableitbar aus  $H_a$  ( $H_a \Rightarrow H_b$ ), wenn es eine Produktion  $p \in P$  und eine Kante  $e$  von  $H_a$  gibt, so daß  $H_a \Rightarrow_e^p H_b$  gilt.

$H_b$  ist in  $G$  ableitbar aus  $H_a$  ( $H_a \Rightarrow^* H_b$ ), wenn es  $H_1, \dots, H_n$ ,  $n \geq 1$ , gibt mit  $H_a = H_1$ ,  $H_n = H_b$  und  $H_{i-1} \Rightarrow H_i$  für  $i \in [2 : n]$ . (Reflexive, transitive Hülle von  $\Rightarrow$ .)

$H_b$  ist in  $G$  in  $m$  Schritten ableitbar aus  $H_a$  ( $H_a \Rightarrow^m H_b$ ), wenn es  $H_1, \dots, H_{m+1}$ ,  $m \geq 1$ , gibt mit  $H_a = H_1$ ,  $H_{m+1} = H_b$  und  $H_{i-1} \Rightarrow H_i$  für  $i \in [2 : m + 1]$ .

Besteht der Hypergraph  $H_a$  aus obiger Definition nur aus einer Hyperkante und den „zugehörigen“ externen Knoten, dann schreiben wir auch:

**Definition 1.11** Sei  $G = (\Sigma, \Delta, P, S)$ ,  $H_a = (V, E, nod, lab, ext)$  mit

$$E = \{e\},$$

$$lab(e) = A \in \Sigma - \Delta,$$

$$V = \{v_1, \dots, v_{rank_\Sigma(A)}\},$$

$$nod(e, i) = v_i \text{ für alle } i \in [1 : rank_\Sigma(A)] \text{ und}$$

$$ext = v_1 \dots v_{rank_\Sigma(A)}.$$

Dann schreiben wir

$$\text{statt } H_a \Rightarrow_e^p H_b \text{ auch } A \Rightarrow^p H_b,$$

$$\text{statt } H_a \Rightarrow H_b \text{ auch } A \Rightarrow H_b,$$

$$\text{statt } H_a \Rightarrow^* H_b \text{ auch } A \Rightarrow^* H_b \text{ und}$$

$$\text{statt } H_a \Rightarrow^m H_b \text{ auch } A \Rightarrow^m H_b.$$

**Definition 1.12 (Satzform)** Sei  $G = (\Sigma, \Delta, P, S)$  eine kontextfreie Hypergraph-Grammatik und  $H$  ein Hypergraph mit  $S \Rightarrow^* H$ .

Dann heißt  $H$  Satzform von  $G$ .

Falls  $H \in HGR(\Delta)$  gilt, wird  $H$  auch Satz von  $G$  genannt.

**Definition 1.13 (Sprache)** Sei  $G = (\Sigma, \Delta, P, S)$  eine kontextfreie Hypergraph-Grammatik, dann ist

$$L(G) = \{H \mid S \Rightarrow^* H \wedge H \text{ ist ein Hypergraph über } \Delta\}$$

die von  $G$  erzeugte (Hypergraph-) Sprache.

Nachdem wir eine Ableitung definiert haben, sind wir jetzt auch in der Lage zu definieren, was eine *reduzierte* Grammatik ist:

**Definition 1.14 (erreichbar)** Sei  $G = (\Sigma, \Delta, P, S)$  eine kontextfreie Hypergraph-Grammatik und  $X \in \Sigma$ .

Dann heißt  $X$  erreichbar genau dann, wenn eine Satzform  $H = (V, E, \text{nod}, \text{lab}, \text{ext})$  von  $G$  und eine Hyperkante  $e \in E$  existieren mit:  $\text{lab}(e) = X$ .

**Definition 1.15 (terminal ableitbar)** Sei  $G = (\Sigma, \Delta, P, S)$  eine kontextfreie Hypergraph-Grammatik und  $X \in \Sigma$ .

Dann heißt  $X$  terminal ableitbar genau dann, wenn  
 $\exists H \in \text{HGR}(\Delta) : X \Rightarrow^* H$

**Definition 1.16 (nutzlos)** Sei  $G = (\Sigma, \Delta, P, S)$  eine kontextfreie Hypergraph-Grammatik und  $p = (X, H) \in P$ ,  $H = (V, E, \text{nod}, \text{lab}, \text{ext})$ .

Die Produktion  $p$  heißt nutzlos genau dann, wenn gilt:

$X$  ist nicht erreichbar oder

$\exists Y \in \Sigma - \Delta, e \in E : \text{lab}(e) = Y \wedge Y$  ist nicht terminal ableitbar.

**Definition 1.17 (reduziert)** Sei  $G = (\Sigma, \Delta, P, S)$  eine kontextfreie Hypergraph-Grammatik.

Dann heißt  $G$  reduziert  $\Leftrightarrow \forall X \in \Sigma : X$  ist erreichbar und  $\forall p \in P : p$  ist nicht nutzlos.

Es gilt:

**Folgerung 1.18** Sei  $G = (\Sigma, \Delta, P, S)$  eine kontextfreie Hypergraph-Grammatik.

Dann existiert eine reduzierte kontextfreie Hypergraph-Grammatik  $G' = (\Sigma', \Delta', P', S')$  mit  $L(G) = L(G')$ .

Die Beweisführung gleicht jener im Falle einer kontextfreien Grammatik – weswegen hier darauf verzichtet wird.

Da also die Beschränkung auf reduzierte Grammatiken keine Einschränkung des Sprachumfanges darstellt, andererseits aber einige Beweise leichter zu führen sind, wenn von reduzierten Grammatiken ausgegangen wird, werden wir in dieser Arbeit bei gegebenen Grammatiken davon ausgehen, daß diese reduziert sind.

Die Begriffe *Pfad* und *Zyklus* entsprechen wohl der intuitiven Vorstellung, die man damit verbindet. Zu beachten ist die implizite Definition der „Richtung“ einer Hyperkante: es wird immer über das Ausgangstentakel zum nächsten Knoten fortgeschritten. Da ausschließlich *gerichtete Pfade* betrachtet werden, wird im weiteren

kurz nur von Pfaden gesprochen - womit dann gerichtete Pfade gemeint sind. Der im folgenden definierte *Elementarpfad* bedeutet anschaulich, daß ein Tentakel zwischen den beiden Netzelementen existiert (dessen Nummer und Richtung angegeben werden). Wir brauchen solche „kurzen“ Pfade, da auch ein Pfad von einem Knoten zu einer Hyperkante (und umgekehrt) definiert werden soll.

**Definition 1.19 (Pfad)** Sei  $H = (V, E, \text{nod}, \text{lab}, \text{ext})$  ein Hypergraph. Falls  $v = \text{nod}(e, \text{rank}(\text{lab}(e)))$ ,  $v \in V$ ,  $e \in E$ , gilt, heißt  $(e, 0, v)$  Elementarpfad. Ebenso heißt  $(v, i, e)$  Elementarpfad, falls  $v = \text{nod}(e, i)$ ,  $i < \text{rank}(\text{lab}(e))$ , gilt.  $(X_1, i_1, X_2, i_2, \dots, X_n)$ ,  $n \geq 2$ ,  $X_j \in V \cup E$ ,  $j \in [1 : n]$ ,  $i_k \in \mathbb{N}_0$ ,  $k \in [1 : n-1]$ , heißt Pfad von der Hyperkante resp. dem Knoten  $X_1$  zur Hyperkante resp. zum Knoten  $X_n$ , falls für alle  $k \in [1 : n-1]$  gilt:  $(X_k, i_k, X_{k+1})$  ist Elementarpfad.

**Bemerkungen:**

1) Offenbar gilt (Bezeichnungen wie in der Definition 1.19):

$$X_k \in V \Rightarrow X_{k+1} \in E,$$

$$X_k \in E \Rightarrow X_{k+1} \in V \wedge i_k = 0 \text{ und}$$

$$X_k \in E \Rightarrow X_{k+1} = \text{nod}(X_k, \text{rank}(\text{lab}(X_k))).$$

2) Somit ist offensichtlich, daß obige Definition eines Pfades eine gewisse Redundanz enthält. Diese aber macht zum einen die Definition kürzer (und hoffentlich verständlicher), zum anderen werden einige Beweise übersichtlicher.

**Definition 1.20 (Zyklus)** Der Hypergraph  $H$  enthält einen Zyklus, wenn es eine Hyperkante  $X_a$  in  $H$  gibt, so daß ein Pfad von  $X_a$  nach  $X_a$  in  $H$  existiert.  $H$  ist zyklensfrei (*acyclic*), wenn  $H$  keinen Zyklus enthält.

Jetzt können wir eine spezielle Teilmenge der Hypergraphen, die Dschungel, definieren. Zur Veranschaulichung folgt am Ende dieses Abschnittes noch ein kurzes Beispiel.

**Definition 1.21 ( $m$ -Dschungel mit  $k$  Variablen)** Ein  $m$ -Hypergraph heißt  $m$ -Dschungel mit  $k$  Variablen,  $m \geq k$ , wenn gilt (sei  $\text{ext} = v_1 \dots v_k \dots v_m$ ):

(a) die ersten  $k$  externen Knoten sind paarweise disjunkt:  $v_i \neq v_j$  für  $i \neq j$ ,  $i, j \in [1 : k]$ ,

(b)  $v_1, \dots, v_k$  besitzen keine Eingangs-Kante,

(c) alle übrigen Knoten sind one-incoming (also alle internen Knoten und  $v_{k+1}, \dots, v_m$  besitzen genau eine Eingangs-Kante),

(d) es treten keine Zyklen und

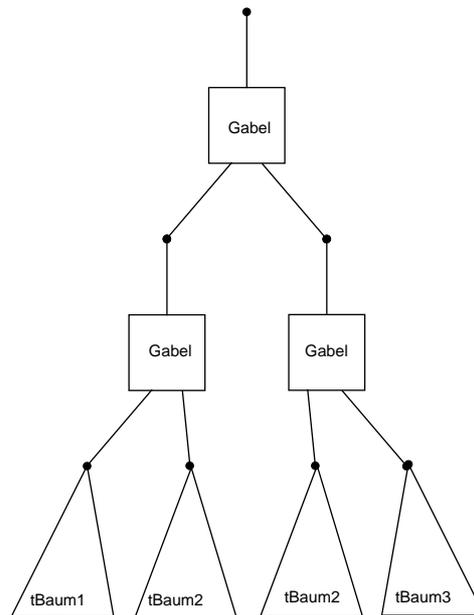


Abbildung 1.2: Skizze eines baumähnlichen Hypergraphen

(e) keine 0-Hyperkanten auf.

Da wir vorwiegend (1-)Dschungel mit 0 Variablen untersuchen werden, definieren wir noch gesondert:

**Definition 1.22 (Dschungel)** Ein 1-Hypergraph heißt Dschungel, wenn gilt:

- (a) alle Knoten sind one-incoming
- (b) es treten keine Zyklen und
- (c) keine 0-Hyperkanten auf.

Ein reiner Dschungel ist ein Dschungel mit der Eigenschaft, daß von jeder Kante (bzw. jedem internen Knoten) ein Pfad zu dem externen (designierten) Knoten führt.

Offensichtlich ist ein Dschungel das gleiche wie ein (1-)Dschungel mit 0 Variablen. Um die Bezeichnung Dschungel zu motivieren, sei folgendes Beispiel gegeben.

**Beispiel 1.23** Als erstes geben wir die Skizze eines Dschungels, der einem Baum ähnlich ist (Abb. 1.2). Dabei ist der Wurzelbereich ausführlich angegeben (alle drei Hyperkanten tragen den Label „Gabel“), die darunterliegenden Teilbäume sind mittels Dreiecke angedeutet. In der Skizze tritt ein Teilbaum zweimal auf. Man beachte, daß, wenn man Hyperkanten vom Rang 2 verwenden würde, eventuell einige Knoten nicht one-incoming wären, somit also dann kein Dschungel vorläge.

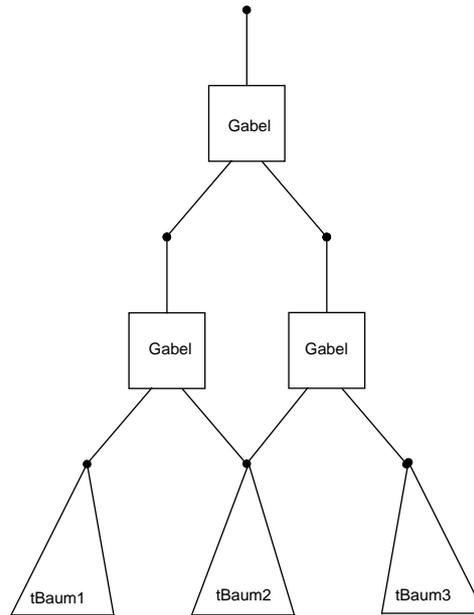


Abbildung 1.3: Skizze eines baumähnlichen Hypergraphen

Nun betrachten wir Abb. 1.3. Der dort dargestellte Dschungel ist dem in Abb. 1.2 dargestellten Baum recht ähnlich, nur der im ersteren Bild zweimal auftretende Teilbaum tritt im zweiten nur einmal auf. Dies kann man so interpretieren, daß Teilbäume wieder zusammenwachsen können bzw. ineinander verschlungen sind – so, wie es in einem realen Dschungel auch vorkommt.

Aus den Bedingungen (b),(c) und (e) kann leicht auf das zahlenmäßige Verhältnis zwischen Knoten und Kanten geschlossen werden:

**Folgerung 1.24** In einem  $m$ -Dschungel  $H = (V, E, nod, lab, ext)$  mit  $k$  Variablen gilt:

$$|V| = |E| + k$$

Speziell also in einem Dschungel ( $k = 0$ ):  $|V| = |E|$

Außerdem lassen sich zwei Aussagen über die Zahl der Hyperkanten, mit denen ein interner Knoten in einem reinen Dschungel  $H$  verbunden ist, machen:

Ein interner Knoten hat *per definitionem* eine Eingangs-Kante (erste Hyperkante), es führt von dem Knoten ein Pfad zu dem externen Knoten von  $H$  (zweite Hyperkante) und wegen Zyklensfreiheit sind diese beiden Hyperkanten nicht identisch:

**Folgerung 1.25** Sei  $H = (V, E, nod, lab, ext)$  ein reiner Dschungel und  $v \in V \setminus ext$ . Dann existieren zwei Hyperkanten  $e_a, e_b \in E$ ,  $e_a \neq e_b$ , so daß  $v \in nod(e_a) \cap nod(e_b)$  gilt.

Wegen der Zyklenfreiheit gilt aber auch (da es mit der Eingangskante des designierten Knotens nur eine „Senke“ gibt):

**Folgerung 1.26** *Sei  $H = (V, E, nod, lab, ext)$  ein reiner Dschungel mit  $|V| \geq 2$ . Dann existieren  $v \in V \setminus ext$  und zwei Hyperkanten  $e_a, e_b \in E, (e_a \neq e_b)$ , so daß  $\forall e_c \in E \setminus \{e_a, e_b\} : v \notin nod(e_c)$  gilt.*

**Bemerkung:**

Bezeichnet man mit  $GRAD(v)$  die Funktion, die die Anzahl der Hyperkanten (nicht der Tentakel!) liefert, mit denen der Knoten  $v$  verbunden ist, dann besagt Folgerung 1.25 also, daß  $GRAD(v) \geq 2$  für alle internen Knoten des reinen Dschungels  $H = (V, E, nod, lab, ext)$  gilt und Folgerung 1.26, daß, falls  $|V| \geq 2$  ist, mindestens ein interner Knoten  $w \in V$  existiert mit  $GRAD(w) \leq 2$ , also  $GRAD(w) = 2$ .

Man kann auch zeigen, daß für  $|V| = n$  und ein  $2 \leq i \leq n$  gilt, daß mindestens  $i - 1$  Knoten  $v_1, \dots, v_{i-1}$  mit  $GRAD(v_j) \leq i, j \in [1 : i - 1]$ , existieren.

Die Dschungel werden in weiterem von besonderem Interesse sein. Dieses ist unter anderem dadurch motiviert, daß bei ihnen den Knoten Bedeutungen zugeordnet werden können, wie man den folgenden Definitionen entnehmen kann. Am Beispiel 1.29 am Ende des Abschnitts ist dargestellt, wie man die Fibonacci-Funktion mit Hypergraphen darstellen kann.

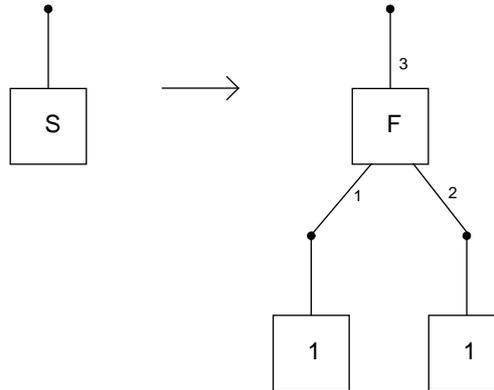
**Definition 1.27 (assoziierter Term)** *Sei  $H = (V, E, nod, lab, ext)$  ein  $m -$  Dschungel mit  $k$  Variablen über einem Marken-Alphabet  $\Sigma$  und  $v \in V$ . Sei  $Y = (y_1, \dots, y_k)$  eine Folge von  $k$  unterschiedlichen Variablen und  $\Delta \subset \Sigma$ . Der mit  $v$  assoziierte Term,  $Term(v, H)$ , ist definiert als:*

$$Term(v, H) = \begin{cases} y_i & \text{falls } v = ext(i), i \in [1, k] \\ lab(e) & \text{falls } e \text{ Eingangskante von} \\ & \text{v ist und } rank_{\Sigma}(e) = 1 \\ lab(e)(Term(nod(e, 1), H), \\ \dots, \\ Term(nod(e, n - 1), H)) & \text{falls } e \text{ Eingangskante von} \\ & \text{v ist und } rank_{\Sigma}(e) = n > 1 \end{cases}$$

Der  $Term(ext(rank(H)), H)$  wird auch als der mit  $H$  assoziierte Term bezeichnet und kurz  $Term(H)$  geschrieben.

**Bemerkung:**

Zwei „unterschiedliche“ Hypergraphen können durchaus den gleichen assoziierten Term besitzen (z. B.  $+(1, 1)$ ). Hierzu haben B. Hoffmann und D. Plump in [HoP188] Untersuchungen durchgeführt.

Abbildung 1.4: Produktion  $p_1$  für Fibonacci

Zwar wird erst im Kapitel 2 die Dschungel-Eigenschaft eingeführt, aber mit dem Hinweis, daß alle durch eine Grammatik mit dieser Eigenschaft zu erzeugenden Hypergraphen Dschungel sind, sei hier schon die Definition der Termsprache angegeben:

**Definition 1.28 (Termsprache  $T(G)$ )** Sei  $G = (\Sigma, \Delta, P, S)$  eine kontextfreie Hypergraph-Grammatik mit Dschungel-Eigenschaft. Dann ist

$$T(G) = \{t \mid \exists H \in L(G) : t = \text{Term}(H)\}.$$

$T(G)$  wird Termsprache von  $G$  genannt. Die Menge aller Termsprachen über kontextfreie Hypergraph-Grammatiken wird mit  $\mathcal{T}_{HG}$  bezeichnet

$$(\mathcal{T}_{HG} = \{T(G) \mid G \text{ ist eine kontextfreie Hypergraph-Grammatik}\}).$$

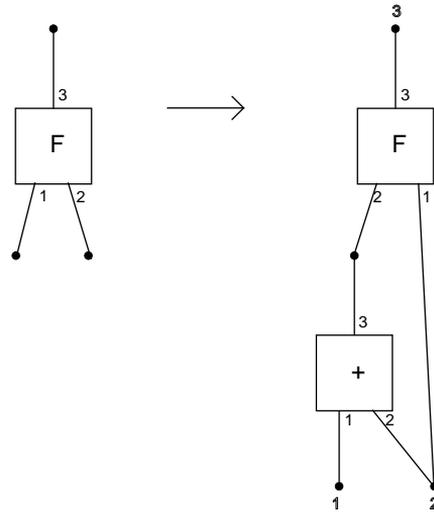
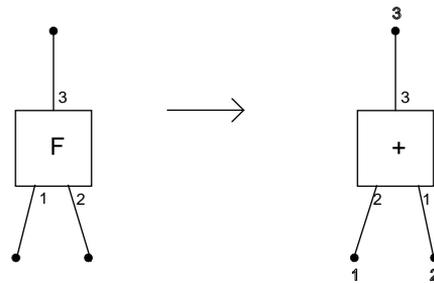
**Beispiel 1.29 (Fibonacci-Funktion)** Sei

$G = (\{S, F, 1, +\}, \{1, +\}, \{p_1, p_2, p_3\}, S)$  eine kontextfreie Hypergraph-Grammatik, wobei die Produktionen wie in den Abbildungen 1.4, 1.5 und 1.6 aussehen.

Dann stellt Abb. 1.7 eine mögliche Ableitung in dieser Grammatik dar. Mit dem designierten Knoten des in der Ableitung am weitesten rechts stehenden Hypergraphen wird der Term  $+(+(1, +(1, 1)), +(1, 1))$  assoziiert. Die Auswertung dieses Terms ergibt den Wert 5. Man überzeuge sich davon, daß für jeden Hypergraphen  $H \in L(G)$  gilt, daß  $\text{Term}(H)$  ein Term ist, dessen Auswertung eine Fibonacci-Zahl liefert. Umgekehrt gibt es zu jeder Fibonacci-Zahl einen Hypergraphen  $H \in L(G)$ , so daß die Auswertung von  $\text{Term}(H)$  gerade diese Fibonacci-Zahl liefert.

## 1.3 L-attributierte Grammatik

Es soll später die Beziehung zwischen speziellen Hypergraph-Grammatiken einerseits und *L-attributierten* Grammatiken andererseits untersucht werden. Daher werden

Abbildung 1.5: Produktion  $p_2$  für FibonacciAbbildung 1.6: Produktion  $p_3$  für Fibonacci

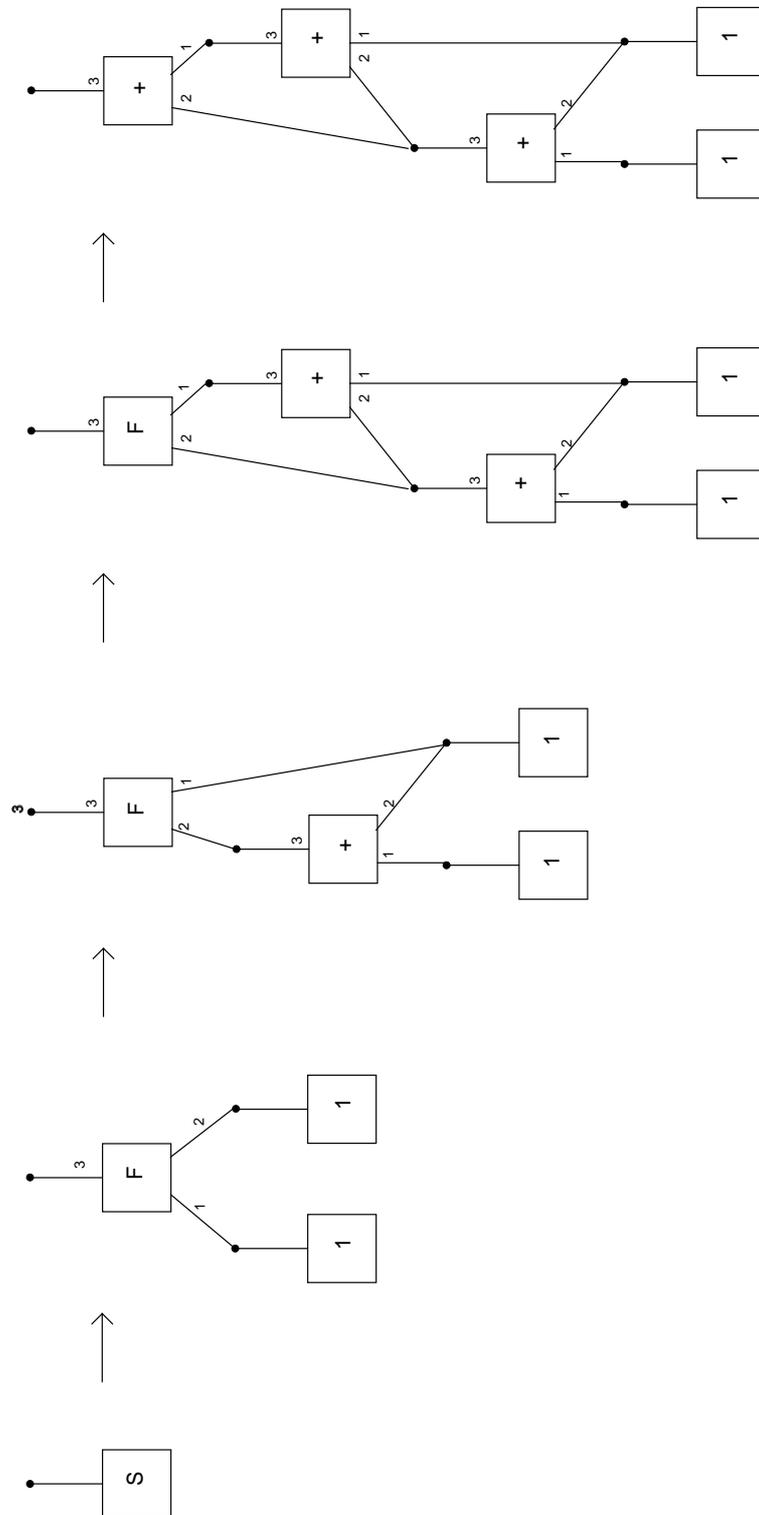


Abbildung 1.7: Beispiel einer Ableitung

hier die notwendigen Definitionen für letztere angeben:

**Definition 1.30 (semantic domain)** Eine semantic domain ist ein Paar  $D = (V, \Gamma)$  mit:  
 $V$  ist eine endliche Menge, die Menge der Werte (value),  
 $\Gamma$  ist ein Marken-Alphabet und  
jedes  $\gamma \in \Gamma$  bezeichnet eine Abbildung  $\gamma_D : V^n \rightarrow V$  mit  $n = \text{rank}_\Gamma(\gamma)$ .

**Definition 1.31 (attributierte Grammatik)** Eine attributierte Grammatik  $G$  besteht aus (1)-(4) wie folgt:

(1) Eine kontextfreie Grammatik  $G_0 = (N_0, T_0, P_0, S_0)$ .

(2) Eine semantic domain  $D = (V, \Gamma)$ .

(3) Eine Attribut-Beschreibung  $(A, \text{Syn}, \text{Inh}, \text{Att})$  mit  $A$  ist eine endliche Menge von Attributen, und  $\text{Syn}$ ,  $\text{Inh}$ , und  $\text{Att}$  sind Abbildungen von  $N_0$  auf  $2^A$ . Für jedes nichtterminale Zeichen  $X \in N_0$  sei  $\text{Syn}(X)$  die Menge der synthetischen Attribute von  $X$  und  $\text{Inh}(X)$  die Menge seiner inheriten Attribute,  $\text{Syn}(X) \cap \text{Inh}(X) = \emptyset$ .  $\text{Att}(X)$  ist die Menge aller Attribute von  $X$ ,  $\text{Att}(X) = \text{Syn}(X) \cup \text{Inh}(X)$ . Die Menge  $\text{Inh}(S_0)$  ist leer, und  $|\text{Syn}(S_0)| = 1$ . Das einzige (synthetische) Attribut von  $S_0$  wird designiertes Attribut von  $G$  genannt und mit  $\alpha_d$  bezeichnet.

(4) Für jede Produktion  $p = X_0 \rightarrow w_0 X_1 w_1 \dots X_n w_n$ ,  $X_i \in N_0, w_i \in T_0^*, i \in [0 : n]$ , existiert eine Menge von Berechnungsvorschriften (semantic rules)  $r_p$ .

Sei  $\text{ins}(p) = \{ \langle \beta, i \rangle \mid (\beta \in \text{Inh}(X_i) \wedge i \in [1 : n]) \vee (\beta \in \text{Syn}(X_0) \wedge i = 0) \}$  die Menge der inside Attribute von  $p$  und  $\text{att}(p) = \{ \langle \beta, i \rangle \mid (\beta \in \text{Att}(X_i), i \in [0 : n]) \}$  die Menge aller Attribute von  $p$ .

Dann enthält  $r_p$  für jedes Attribut  $\langle \alpha, j \rangle \in \text{ins}(p)$  eine Berechnungsvorschrift der Form  $\langle \alpha, j \rangle = t$  mit  $t$  ist ein Term über  $\text{att}(p) \cup \Gamma$ .

$r_p$  enthalte keine weiteren Berechnungsvorschriften.

Betrachtet man einen Ableitungsbaum eines Wortes  $w \in L(G_0)$  und wendet die den Produktionen zugehörigen Berechnungsvorschriften an, so läßt sich (unter Umständen) der Wert des designierten Attributes berechnen. Dieser wird auch als Bedeutung von  $w$  aufgefaßt.

Der Wert des designierten Attributes läßt sich allerdings eventuell nicht berechnen, wenn die Anwendung der Berechnungsvorschriften zu Zirkelschlüssen führt. Eine attributierte Grammatik, bei der solches ausgeschlossen ist, wird *zyklenfrei* genannt.

Mit weiteren Einschränkungen erhält man:

**Definition 1.32 (L-attributierte Grammatik)** Eine attributierte Grammatik  $G$  heißt L-attributierte Grammatik, wenn sie außerdem folgende Bedingungen erfüllt:

(5)  $\forall X \in N : |\text{Syn}(X)| = 1$ .

(6) Für die Berechnung eines inheriten Attributes wird nur der Teil der Attribute aus  $att(p)$  benutzt, der zu weiter links stehenden Nichtterminalen gehört:

Sei  $att_j(p) = \{ \langle \beta, i \rangle \mid (\beta \in Att(X_i), i \in [1 : (j - 1)]) \}$ .

Ist  $\langle \alpha, j \rangle = t$  in  $r_p$  und  $\langle \alpha, j \rangle$  inherites Attribut von  $X_j$ , dann ist  $t$  ein Term über  $att_j(p) \cup Inh(X_0) \cup \Gamma$ .

Offenbar ist eine L-attributierte Grammatik zyklensfrei.

Wenn man einen Ableitungsbaum eines Wortes  $w \in L(G_0)$  kennt, so kann man bei Linksableitung jeweils für das nächste abzuleitende Symbol alle inheriten Attributwerte bestimmen und, wenn dieses schließlich terminal abgeleitet wurde, auch den Wert des synthetischen Attributes.

Dies ist in der Praxis deshalb von besonderem Interesse, weil bei der Top-Down-Analyse gerade dieser Ableitungsbaum in passender Reihenfolge konstruiert wird und man daher parallel zur Analyse die Berechnung des designierten Attributes vornehmen kann.

**Definition 1.33 (lhs(p))** Sei  $G$  eine L-attributierte Grammatik,

$G_0 = (N_0, T_0, P_0, S_0)$  die kontextfreie Grammatik von  $G$  und  $p : A \rightarrow w$  eine Produktion aus  $G_0$ . Dann ist

$lhs(p) = A$  (left-hand side).

**Definition 1.34 (Termsprache  $T_{LaG}(G)$ )** Sei  $G$  eine L-attributierte Grammatik.

Dann ist

$T_{LaG}(G) = \{ t \mid \text{es existiert ein Ableitungsbaum zu } w \in L(G), \text{ der } t \text{ als Term zur Berechnung des designierten Attributes festlegt} \}$ .

$T_{LaG}(G)$  wird Termsprache von  $G$  genannt. Die Menge aller Termsprachen über L-attributierte Grammatiken wird mit  $\mathcal{T}_{LaG}$  bezeichnet

( $\mathcal{T}_{LaG} = \{ T(G) \mid G \text{ ist eine L-attributierte Grammatik} \}$ ).

## 1.4 IO-Makrogrammatik

Eine weitere Möglichkeit zur Erzeugung von Termen bzw. Worten stellen IO-Makrogrammatiken dar, die hier kurz vorgestellt werden sollen. Auch diese werden später in Bezug zu Hypergraph-Grammatiken gesetzt.

**Definition 1.35 (Makro-Term)** Sei  $\Sigma$  eine Menge von terminalen Zeichen und  $\mathcal{F}$  eine Menge von Funktionssymbolen, die von ersterer disjunkt ist ( $\Sigma \cap \mathcal{F} = \emptyset$ ).

Für alle Funktionssymbole  $F \in \mathcal{F}$  wird die Anzahl der Argumente von  $F$  mit  $\rho(F)$  bezeichnet. Außerdem seien die Sonderzeichen “(, “)” und “,“ in keiner der beiden

Mengen enthalten ( $\notin (\Sigma \cup \mathcal{F}) \dots$ ).

Ein Makro-Term über  $\Sigma, \mathcal{F}, \rho$  ist ein Wort über  $\Sigma \cup \mathcal{F} \cup \{(\} \cup \{,\}$ , das folgende Bedingungen erfüllt:

- (a) Das leere Wort  $\epsilon$  ist ein Makro-Term. Falls  $a \in \Sigma$  ist, dann ist  $a$  ein Makro-Term.
- (b) Falls  $\phi_1$  und  $\phi_2$  Makro-Terme sind, dann ist  $\phi_1\phi_2$  ein Makro-Term.
- (c) Falls  $F \in \mathcal{F}$  ist und  $\sigma_1, \dots, \sigma_{\rho(F)}$  Makro-Terme sind, dann ist  $F(\sigma_1, \dots, \sigma_{\rho(F)})$  ein Makro-Term.
- (d) Ein Wort über  $\Sigma \cup \mathcal{F} \cup \{(\} \cup \{,\}$  ist ein Makro-Term, wenn es durch endlich viele Anwendungen der Regeln (a), (b) und (c) konstruiert werden kann.

**Bemerkungen:**

1) Der unter (b) beschriebene „Konstruktionsschritt“ zur Bildung eines Makro-Terms stellt im Prinzip eine spezielle Funktion dar. Es wäre ebenso möglich gewesen, auf diese Konstruktionsregel zu verzichten und  $\mathcal{F}$  um ein Konkatinationssymbol  $\circ$  mit  $\rho(\circ) = 2$  zu erweitern. Die Berechnung des Wertes muß dann als  $\circ(a, b) = ab$  definiert werden.

2) Ein Makro-Term ist offensichtlich immer wohlgeklammert, d.h. die Anzahl der öffnenden Klammern ist gleich der Anzahl der schließenden und für jedes Praefix gilt, daß die Anzahl der schließenden Klammern nicht größer als die der öffnenden ist.

Jetzt können wir eine Makrogrammatik ähnlich wie in [DuPaSeSp] definieren:

**Definition 1.36 (IO-Makrogrammatik)** Eine Makrogrammatik (macrogrammar) ist ein 6-Tupel  $G_M = (\Sigma, \mathcal{F}, \mathcal{V}, \rho, S, \Pi)$  mit

- $\Sigma$  ist eine endliche nichtleere Menge von terminalen Zeichen,
- $\mathcal{F}$  ist eine endliche nichtleere Menge von Funktionssymbolen,
- $\mathcal{V}$  ist eine endliche Menge von Variablennamen,
- $\rho$  ist eine Funktion, die  $\mathcal{F}$  auf nichtnegative ganze Zahlen abbildet,
- $S \in \mathcal{F}$  ist das Startsymbol mit  $\rho(S) = 0$ , und
- $\Pi$  ist eine endliche Menge von (Makro-) Produktionen.

Die Mengen  $\Sigma, \mathcal{F}$  und  $\mathcal{V}$  seien paarweise disjunkt. Die Produktionen in  $\Pi$  seien von der Form

$$F(x_1, \dots, x_{\rho(F)}) \rightarrow \theta$$

mit  $F \in \mathcal{F}$ ,  $x_i \in \mathcal{V}$  mit  $x_i \neq x_j$  für  $i \neq j$ , und  $\theta$  ist ein Makro-Term über  $\Sigma \cup \{x_1, \dots, x_{\rho(F)}\}, \mathcal{F}, \rho$ .

Eine IO-Makrogrammatik ist eine Makrogrammatik, bei der zusätzlich festgelegt ist, daß bei der Ableitung nur Funktionssymbole abgeleitet werden dürfen, in deren „Parameterlisten“ keine Funktionssymbole auftauchen (es ist von innen nach außen abzuleiten).

Bei einer IO-Makrogrammatik werden die Makro-Produktionen immer auf die innersten Funktionssymbole angewandt, d. h., die Argumente der Funktion bestehen nur aus terminalen Symbolen. Die Ableitung erfolgt also von innen nach außen (inside-out).

Für unsere Zwecke ist eine Variante der obigen Makrogrammatik günstiger, die im folgenden vorgestellt wird:

**Definition 1.37 (termgenerierende IO-Makrogrammatik)**

Eine termgenerierende Makrogrammatik ist ein 6-Tupel  $G_M = (\Sigma, \mathcal{F}, \mathcal{V}, \rho, S, \Pi)$  mit

- $\Sigma$  ist eine endliche nichtleere Menge von terminalen Zeichen,
- $\mathcal{F}$  ist eine endliche nichtleere Menge von Funktionssymbolen,
- $\mathcal{V}$  ist eine endliche Menge von Variablennamen,
- $\rho$  ist eine Funktion, die  $\mathcal{F} \cup \Sigma$  auf nichtnegative ganze Zahlen abbildet,
- $S \in \mathcal{F}$  ist das Startsymbol mit  $\rho(S) = 0$ , und
- $\Pi$  ist eine endliche Menge von (Makro-) Produktionen.

Die Mengen  $\Sigma, \mathcal{F}$  und  $\mathcal{V}$  seien paarweise disjunkt. Die Produktionen in  $\Pi$  seien von der Form

$$F(x_1, \dots, x_{\rho(F)}) \rightarrow \theta$$

mit  $F \in \mathcal{F}$ ,  $x_i \in \mathcal{V}$  mit  $x_i \neq x_j$  für  $i \neq j$ , und  $\theta$  ist ein Term über  $\Sigma \cup \{x_1, \dots, x_{\rho(F)}\}, \mathcal{F}, \rho$ .

Eine termgenerierende IO-Makrogrammatik ist eine termgenerierende Makrogrammatik, bei der zusätzlich festgelegt ist, daß bei der Ableitung nur Funktionssymbole abgeleitet werden dürfen, in deren „Parameterlisten“ keine Funktionssymbole auftauchen (es ist von innen nach außen abzuleiten).

Die Unterschiede zur „normalen“ Makrogrammatik liegen in der Definition darin, daß erstens  $\rho$  auch für die terminalen Zeichen definiert ist und zweitens die rechten Seiten der Produktionen Terme (statt Makro-Terme) sind. Diese Unterschiede

bewirken, daß die Sätze, die mit einer termgenerierenden Makrogrammatik „produziert“ werden können, Terme sind – wie der Name schon verspricht. Im anderen Falle ergeben sich Worte über dem Alphabet  $\Sigma$ .

Die beiden hier definierten Formen von IO-Makrogrammatiken sind gleich mächtig, was hier nur kurz angedeutet werden soll:

Gegeben eine termgenerierende IO-Makrogrammatik erhält man die „passende“ IO-Makrogrammatik dadurch, daß man  $\Sigma$  um die Zeichen “(“, “,” und “)” erweitert und alles übrige übernimmt - wobei in geeigneter Weise unterschieden werden muß zwischen einerseits Klammern und Kommata, die zu terminalen Zeichen gehören, und andererseits jenen, die zu nichtterminalen Zeichen gehören.

Umgekehrt muß man die Worte auf den rechten Seiten einer Makrogrammatik in Terme „verwandeln“, indem man rekursiv jede Konkatenation  $t_1t_2$  in  $\circ(t_1, t_2)$  umändert, wobei  $\circ$  ein neues Zeichen ist. (Wegen der Assoziativität der Konkatenation ist es egal, ob  $t_1t_2t_3$  in  $\circ(t_1, \circ(t_2, t_3))$  oder  $\circ(\circ(t_1, t_2), t_3)$  umgewandelt wird.)

$\circ$  hat den Rang 2 und gehört zu den terminalen Zeichen, allen übrigen terminalen Zeichen wird der Rang 0 zugeordnet.

Die so erhaltene termgenerierende Makrogrammatik erzeugt Terme, deren Auswertung –  $\circ(t_1, t_2) = t_1t_2$  – genau die Worte der vorgelegten Makrogrammatik liefert.

**Definition 1.38 (lhs(p))** Sei  $G_M = (\Sigma, \mathcal{F}, \mathcal{V}, \rho, S, \Pi)$  eine termgenerierende Makrogrammatik und  $F(x_1, \dots, x_{\rho(F)}) \rightarrow \theta$  eine Produktion aus  $\Pi$ . Dann ist  $lhs(p) = F$  (left-hand side, ohne Parameterliste!).

**Definition 1.39 (Termsprache  $T_{IOM}(G)$ )** Sei  $G = (\Sigma, \mathcal{F}, \mathcal{V}, \rho, S, \Pi)$  eine termgenerierende IO-Makrogrammatik. Dann ist

$$T_{IOM}(G) = L(G).$$

$T_{IOM}(G)$  wird Termsprache von  $G$  genannt.

Die Menge aller Termsprachen über termgenerierende IO-Makrogrammatiken wird mit  $\mathcal{T}_{IOM}$  bezeichnet

$$(\mathcal{T}_{IOM} = \{T_{IOM}(G) | G \text{ ist eine termgenerierende IO-Makrogrammatik}\}).$$

## 1.5 Ordnung muß sein

In diesem Abschnitt sollen Relationen unter den Hyperkanten eines Hypergraphen vorgestellt werden, die sich dann als (partielle oder totale) Ordnungen erweisen werden, wenn der Hypergraph ein (reiner) Dschungel ist.

Zunächst benötigen wir den allgemeinen Ordnungsbegriff:

**Definition 1.40 (Ordnung)** Eine Relation  $R \subseteq A \times A$  heißt strenge partielle Ordnung  $\Leftrightarrow$

a)  $R$  ist irreflexiv ( $(a, a) \notin R$ )

b)  $R$  ist transitiv ( $(a, b) \in R \wedge (b, c) \in R \Rightarrow (a, c) \in R$ )

$R$  heißt strenge totale Ordnung  $\Leftrightarrow$

$R$  ist strenge partielle Ordnung mit  $\forall a \forall b \neq a : (a, b) \in R \vee (b, a) \in R$ .

Statt strenge partielle Ordnung sagen wir auch kürzer partielle Ordnung und statt strenge totale Ordnung auch kurz nur Ordnung.

**Bemerkung:**

Aus den Bedingungen a) und b) folgt als weitere Eigenschaft:

c)  $R$  ist antisymmetrisch ( $(a, b) \in R \Rightarrow (b, a) \notin R$ ).

Man kann eine Ordnung auf einer endlichen Menge definieren, indem man sie explizit angibt (z. B. durch Aufzählung). Wesentlich nützlicher ist es aber meist, wenn man stattdessen ein Ordnungsprinzip angeben kann. Zum einen kann man dann beim Vergleich zweier Elemente das Ordnungsprinzip direkt anwenden. Zum anderen kann man so auch für unendliche Mengen und deren Teilmengen Ordnungen definieren (z. B. alphabetische Ordnung in Lexika).

**Definition 1.41 (Vorgängerrelation)** Es sei  $H = (V, E, \text{nod}, \text{lab}, \text{ext})$  ein Hypergraph.

$R \subset E \times E$  heißt Vorgängerrelation, falls für alle  $(X_a, X_b) \in E \times E$  gilt:  $(X_a, X_b) \in R \Leftrightarrow$  es existiert einen Pfad von  $X_a$  nach  $X_b$ .

Für  $(X_a, X_b) \in R$  schreiben wir auch  $X_a <_v X_b$ .

**Satz 1.42** In einem zyklensfreien Hypergraphen stellt die Vorgängerrelation eine partielle Ordnung dar.

**Beweis:**

Sei  $H = (V, E, \text{nod}, \text{lab}, \text{ext})$  ein zyklensfreier Hypergraph und  $X_a, X_b, X_c \in E$  Hyperkanten von  $H$ .

a) Wegen Zyklensfreiheit gibt es keinen Pfad von  $X_a$  nach  $X_a$  (also  $X_a \not<_v X_a$ ).

b)  $X_a <_v X_b \wedge X_b <_v X_c \Rightarrow$  es existiert ein Pfad von  $X_a$  nach  $X_b$  und einer von  $X_b$  nach  $X_c \Rightarrow$  es existiert ein Pfad von  $X_a$  nach  $X_c \Rightarrow X_a <_v X_c$ .

□

**Bemerkungen:**

1) Es gilt auch die Umkehrung des Satzes.

2) Die Vorgängerrelation wird im weiteren auch  $v$ -Ordnung genannt.

Als nächstes wird die  $vl$ -Ordnung eingeführt, die eine Einbettung der  $v$ -Ordnung darstellt.

Um die Definition zu entzerren, definieren wir zunächst:

**Definition 1.43 (linkslaufend)** Gegeben seien der Dschungel

$H = (V, E, \text{nod}, \text{lab}, \text{ext})$  und zwei Pfade von  $A$  bzw.  $B$  nach  $C$ .

Seien  $p_a = (A = A_0, 0, v_0, i_0, A_1, \dots, A_m = C)$ ,  $m \geq 1$  und  $p_b = (B = B_0, 0, w_0, j_0, B_1, \dots, B_n = C)$ ,  $n \geq 1$  diese Pfade.

Wenn es ein  $1 \leq k \leq \min\{m, n\}$  gibt, so daß

$i_{m-l} = j_{n-l}$ ,  $l \in [0 : k - 1]$ , und  $i_{m-k} \neq j_{n-k}$  gilt,

dann heißt der Pfad  $p_a$  linkslaufend vom Pfad  $p_b$ , falls  $i_{m-k} < j_{n-k}$  gilt (und umgekehrt).

**Bemerkungen:**

1) Wegen der one-incoming-Eigenschaft gilt:  $A_{m-l} = B_{n-l}$ ,  $l \in [0 : k - 1]$  und  $A_{m-k} \neq B_{n-k}$ .

Betrachtet man nämlich von  $C$  ausgehend die Pfade rückwärts, so legt  $i_m$  bzw.  $j_n$  das Tentakel fest, über welches der letzte Knoten des Pfades ( $v_{m-1}$  bzw.  $w_{n-1}$ ) zu erreichen ist. Gilt  $i_m = j_n$ , dann gilt auch  $v_{m-1} = w_{n-1}$  und wegen der Eindeutigkeit der Eingangskanten  $A_{m-1} = B_{n-1}$ . Für diese Hyperkante kann dann ganz entsprechend argumentiert werden.

2) In einem reinen Dschungel mit  $X_r$  ( $r$  für *root*) als Eingangs-Kante von  $\text{ext}(1)$  gibt es von jeder anderen Hyperkante einen Pfad zu  $X_r$ .

3) Wenn einer der beiden Pfade Teil des anderen ist oder die beiden sogar identisch sind, ist keiner zu dem anderen linkslaufend (weil es kein  $k$  mit den geforderten Eigenschaften gibt).

**Satz 1.44** Sei  $H = (V, E, \text{nod}, \text{lab}, \text{ext})$  ein Dschungel und  $X_l \in E$  eine Hyperkante von  $H$ .

Sei  $P_l = \{p = (X_0, 0, v_0, x_0, X_1, \dots, X_t) \mid X_i, X_t \in E, x_i \in \mathbb{N}, v_i \in V, i \in [0 : t - 1], t \geq 1 \wedge p \text{ ist ein Pfad und } X_t = X_l\}$ .

Sei  $R \subset P_l \times P_l$ , so daß für alle  $(p_1, p_2) \in P_l \times P_l$  gilt :  $(p_1, p_2) \in R \Leftrightarrow p_1$  ist linkslaufend von  $p_2$ .

Dann legt  $R$  eine partielle Ordnung fest.

**Beweis:**

a) (irreflexiv) Ein Pfad ist nicht zu sich selbst linkslaufend.

b) (transitiv) Seien  $p_a = (X_a = A_0, 0, v_0^a, a_0, A_1, \dots, A_m = X_l)$ ,  $A_i \in E$ ,  $a_i \in \mathbb{N}$ ,  $v_i^a \in V$ ,  $i \in [0 : m - 1]$ ,  $p_b = (X_b = B_0, 0, v_0^b, b_0, B_1, \dots, B_n = X_l)$ ,  $B_i \in E$ ,  $b_i \in \mathbb{N}$ ,  $v_i^b \in V$ ,  $i \in [0 : n - 1]$  und  $p_c = (X_c = C_0, 0, v_0^c, c_0, C_1, \dots, C_s = X_l)$ ,  $C_i \in E$ ,  $c_i \in \mathbb{N}$ ,  $v_i^c \in V$ ,  $i \in [0 : s - 1]$  Pfade von  $H$  und es gelte:

$p_a$  ist linkslaufend von  $p_b$  und  $p_b$  ist linkslaufend von  $p_c$ .

Da  $p_a$  linkslaufend zu  $p_b$  ist, gibt es ein  $k \geq 1$  mit  $a_{m-i} = b_{n-i}$ ,  $i \in [1 : k - 1]$  und  $a_{m-k} < b_{n-k}$ .

Da  $p_b$  linkslaufend zu  $p_c$  ist, gibt es ein  $l \geq 1$  mit  $b_{n-i} = c_{s-i}$ ,  $i \in [1 : l - 1]$  und  $b_{n-l} < c_{s-l}$ .

Nun sind drei Fälle zu unterscheiden:

- 1)  $k > l \Rightarrow a_{m-i} = b_{n-i} = c_{s-i}, i \in [1 : l - 1]$   
 $\wedge a_{m-l} = b_{n-l} < c_{s-l}$   
 $\Rightarrow p_a$  ist linkslaufend von  $p_c$ .
- 2)  $k = l \Rightarrow a_{m-i} = b_{n-i} = c_{s-i}, i \in [1 : l - 1]$   
 $\wedge a_{m-l} < b_{n-l} < c_{s-l}$   
 $\Rightarrow p_a$  ist linkslaufend von  $p_c$ .
- 3)  $k < l \Rightarrow a_{m-i} = b_{n-i} = c_{s-i}, i \in [1 : k - 1]$   
 $\wedge a_{m-k} < b_{n-k} = c_{s-k}$   
 $\Rightarrow p_a$  ist linkslaufend von  $p_c$ .

Also folgt aus  $p_a$  linkslaufend zu  $p_b$  und  $p_b$  linkslaufend zu  $p_c$ , daß  $p_a$  linkslaufend zu  $p_c$  ist und damit die Transitivität.

□

**Definition 1.45 (Links-Relation)** *Es sei  $H = (V, E, nod, lab, ext)$  ein Dschungel mit  $X_r$  als Eingangs-Kante von  $ext(1)$ .*

*$R \subset E \times E$  heißt Links-Relation, falls für alle  $(X_a, X_b) \in E \times E$  gilt:  $(X_a, X_b) \in R \Leftrightarrow$  es existiert mindestens ein Pfad von  $X_b$  zu  $X_r$  und es existiert ein Pfad  $p_a$  von  $X_a$  nach  $X_r$ , so daß für jeden Pfad  $p_b$  von  $X_b$  nach  $X_r$  gilt:  $p_a$  ist linkslaufend  $p_b$ .*

*Für  $(X_a, X_b) \in R$  schreiben wir auch  $X_a <_l X_b$ .*

**Bemerkung:**

Hyperkanten, von denen aus kein Pfad zu  $X_r$  führt (z.B.  $X_r$ , da Dschungel zyklenfrei und Pfade nicht „leer“ sind), tauchen weder als erste noch als zweite Komponente in  $R$  auf.

**Satz 1.46** *In einem Dschungel stellt die Links-Relation eine partielle Ordnung dar.*

**Beweis:**

Sei  $H = (V, E, nod, lab, ext)$  der Dschungel und  $X_r \in E$  die Eingangs-Kante von  $ext(1)$ .

a) (irreflexiv) Ein Pfad ist nicht zu sich selbst linkslaufend, also gibt es auch keinen Pfad von einer Hyperkante  $X_a \in E$  zu  $X_r$ , der zu allen Pfaden von  $X_a$  zu  $X_r$  linkslaufend ist (und somit kein  $X_a$  mit  $X_a <_l X_a$ ).

Wenn kein Pfad von  $X_a$  zu  $X_r$  existiert, folgt ebenfalls  $X_a \not<_l X_a$ .

b) (transitiv) Seien  $X_a, X_b, X_c \in E$  und gelte  $X_a <_l X_b$  und  $X_b <_l X_c$ .

Es existiert ein Pfad  $p_a$ , der linkslaufend zu allen Pfaden von  $X_b$  zu  $X_r$  ist, und ein Pfad  $p_b$ , der linkslaufend zu allen Pfaden von  $X_c$  zu  $X_r$  ist.

Da linkslaufend transitiv ist (Satz 1.44) folgt, daß  $p_a$  linkslaufend zu allen Pfaden von  $X_c$  zu  $X_r$  ist und damit  $X_a <_l X_c$ .

□

Zwischen der Vorgängerrelation und der Links-Relation besteht eine gewisse Verträglichkeit, die im nächsten Satz zum Ausdruck kommt:

**Satz 1.47** Sei  $H = (V, E, \text{nod}, \text{lab}, \text{ext})$  ein Dschungel und  $X_a, X_b \in E$  zwei Hyperkanten von  $H$ . Dann gilt:

$$X_a <_v X_b \Rightarrow X_b \not<_l X_a$$

**Beweis:**

Annahme des Gegenteils: Es gelte  $X_a <_v X_b$  und  $X_b <_l X_a$  in  $H$ .

Aus  $X_a <_v X_b$  folgt, daß es einen Pfad  $p_a = (X_a = A_0, 0, v_0^a, a_0, A_1, \dots, A_m = X_b)$ ,  $A_i \in E$ ,  $a_i \in \mathbb{N}$ ,  $v_i^a \in V$ ,  $i \in [0 : m - 1]$  gibt.

Aus  $X_b <_l X_a$  folgt, daß ein Pfad  $p_b = (X_b = B_0, 0, v_0^b, b_0, B_1, \dots, B_n = X_r)$ ,  $B_i \in E$ ,  $b_i \in \mathbb{N}$ ,  $v_i^b \in V$ ,  $i \in [0 : n - 1]$  existiert, der linkslaufend von allen Pfaden von  $X_a$  nach  $X_r$  ist.

Betrachte  $p_{ab} = (X_a = A_0, 0, v_0^a, a_0, A_1, \dots, A_m = X_b = B_0, 0, v_0^b, b_0, B_1, \dots, B_n = X_r)$ . Dies ist ein Pfad von  $X_a$  nach  $X_r$ , und es gilt:  $p_b$  ist nicht linkslaufend von  $p_{ab}$ , also Widerspruch zu obiger Annahme. □

**Definition 1.48 (Vorgänger-Links-Relation)** Es sei  $H = (V, E, \text{nod}, \text{lab}, \text{ext})$  ein (1-)Hypergraph.

$R \subset E \times E$  heißt Vorgänger-Links-Relation, falls für alle  $(X_a, X_b) \in E \times E$  gilt:  $(X_a, X_b) \in R \Leftrightarrow X_a <_v X_b \vee X_a <_l X_b$ .

Für  $(X_a, X_b) \in R$  schreiben wir auch  $X_a <_{vl} X_b$ .

**Satz 1.49** In einem reinen Dschungel stellt die Vorgänger-Links-Relation eine totale Ordnung dar.

**Beweis:**

Sei  $H = (V, E, \text{nod}, \text{lab}, \text{ext})$  der Dschungel und  $X_r$  die Eingangs-Kante von  $\text{ext}(1)$ .

a) (irreflexiv) Annahme: Es gibt ein  $X \in E$  mit  $X <_{vl} X$ , daraus folgt:

$X <_v X \vee X <_l X$ , ersteres kann wegen Satz 1.42, zweiteres wegen Satz 1.46 nicht gelten, also Widerspruch!

b) (transitiv) Seien  $X_a, X_b, X_c \in E$  Hyperkanten von  $H$  mit  $X_a <_{vl} X_b$  und  $X_b <_{vl} X_c$ . Daraus folgt  $(X_a <_v X_b \vee X_a <_l X_b) \wedge (X_b <_v X_c \vee X_b <_l X_c)$ .

Dann sind die vier Fälle zu unterscheiden:

$$1) X_a <_v X_b \wedge X_b <_v X_c \Rightarrow (\text{wegen Satz 1.42}) X_a <_v X_c \Rightarrow X_a <_{vl} X_c$$

$$2) X_a <_v X_b \wedge X_b <_l X_c \Rightarrow \text{es existiert ein Pfad } p_{ab} = (X_a = A_0, 0, v_0^a, a_0, A_1, \dots, A_m = X_b), A_i \in E, a_i \in \mathbb{N}, v_i^a \in V, i \in [0 : m - 1] \text{ und ein Pfad } p_{br} = (X_b = B_0, 0, v_0^b, b_0, B_1, \dots, B_n = X_r), B_i \in E, b_i \in \mathbb{N}, v_i^b \in V$$

$V$ ,  $i \in [0 : n - 1]$ , der linkslaufend zu allen Pfaden von  $X_c$  zu  $X_r$  ist.

Sei  $p_{abr} = (X_a = A_0, 0, v_0^a, a_0, A_1, \dots, A_m = X_b = B_0, 0, v_0^b, b_0, B_1, \dots, B_n = X_r)$ .

Offenbar ist  $p_{abr}$  linkslaufend zu allen Pfaden von  $X_c$  zu  $X_r$  (falls solche existieren). Es folgt:  $X_a <_l X_c \Rightarrow X_a <_{vl} X_c$ .

3)  $X_a <_l X_b \wedge X_b <_v X_c$ . Hier muß unterschieden werden, ob  $X_a <_v X_c$  gilt oder nicht:

a) Es gilt  $X_a <_v X_c$ , dann liegt Fall 1) vor und es folgt  $X_a <_{vl} X_c$ .

b) Es gilt  $X_a \not<_v X_c$ . Es existiert (wegen  $X_a <_l X_b$ ) ein Pfad  $p_{ar} = (X_a = A_0, 0, v_0^a, a_0, A_1, \dots, A_m = X_r)$ ,  $A_i \in E$ ,  $a_i \in \mathbb{N}$ ,  $v_i^a \in V$ ,  $i \in [0 : m - 1]$ , der linkslaufend zu allen Pfaden von  $X_b$  zu  $X_r$  ist, und (wegen  $X_b <_v X_c$ ) ein Pfad  $p_{bc} = (X_b = B_0, 0, v_0^b, b_0, B_1, \dots, B_n = X_c)$ ,  $B_i \in E$ ,  $b_i \in \mathbb{N}$ ,  $v_i^b \in V$ ,  $i \in [0 : n - 1]$ .

Sei  $p_c = (X_c = C_0, 0, v_0^c, c_0, C_1, \dots, C_s = X_r)$ ,  $C_i \in E$ ,  $c_i \in \mathbb{N}$ ,  $v_i^c \in V$ ,  $i \in [0 : s - 1]$  ein Pfad von  $X_c$  nach  $X_r$ .

Betrachte  $p_{bcr} = (X_b = B_0, 0, v_0^b, b_0, B_1, \dots, B_n = X_c = C_0, 0, v_0^c, c_0, C_1, \dots, C_s = X_r)$ . Wegen  $X_a <_l X_b$  ist  $p_{ar}$  linkslaufend von  $p_{bcr}$  und es kann, da  $X_a \not<_v X_c$  gilt, gefolgert werden, daß  $p_{ar}$  auch linkslaufend von  $p_{cr}$  ist. Da dieses für jeden Pfad von  $X_c$  nach  $X_r$  gilt, folgt  $X_a <_l X_c$  und damit  $X_a <_{vl} X_c$ .

4)  $X_a <_l X_b \wedge X_b <_l X_c \Rightarrow$  (wegen Satz 1.46)  $X_a <_l X_c \Rightarrow X_a <_{vl} X_c$ .

Also folgt in jedem Fall  $X_a <_{vl} X_c$  und damit die Transitivität der Vorgänger-Links-Relation.

Bleibt die Totalität zu zeigen:

Seien  $X_a, X_b \in E$  zwei Hyperkanten von  $H$ .

1) Falls  $X_a <_v X_b$  oder  $X_b <_v X_a$  gilt, folgt  $X_a <_{vl} X_b$  bzw.  $X_b <_{vl} X_a$ .

2) Ansonsten gilt, da  $H$  ein reiner Dschungel ist, daß mindestens je ein Pfad von  $X_a$  bzw.  $X_b$  nach  $X_r$  existiert. Betrachtet man jetzt die (endlichen) Mengen  $P_a$ , die Menge der Pfade von  $X_a$  nach  $X_r$  und  $P_b$ , die Menge der Pfade von  $X_b$  nach  $X_r$ , so sind beide nicht leer. Außerdem gilt, wegen  $X_a \not<_v X_b$  und  $X_b \not<_v X_a$ , für je zwei voneinander verschiedene Pfade  $p_1$  und  $p_2$  aus der Vereinigungsmenge, daß entweder  $p_1$  linkslaufend von  $p_2$  ist oder umgekehrt. Zusammen mit Satz 1.44 ergibt sich also, daß linkslaufend eine totale Ordnung unter den Elementen der Vereinigungsmenge darstellt. Also läßt sich ein (bzgl. linkslaufend) kleinstes Element  $p_k$  aus der Vereinigungsmenge bestimmen.

Falls  $p_k \in P_a$  ist, dann ist  $p_k$  linkslaufend zu allen Pfaden von  $X_b$  nach  $X_r$  und somit  $X_a <_{vl} X_b$ , sonst gilt  $X_b <_{vl} X_a$ . Also gilt in jedem Fall entweder  $X_a <_{vl} X_b$  oder  $X_b <_{vl} X_a$ , womit gezeigt ist, daß die Vorgänger-Links-Relation total ist.

□

### Bemerkungen:

- 1) Die Vorgänger-Links-Relation wird im weiteren auch  $vl$ -Ordnung genannt.
- 2) Das größte Element dieser Ordnung ist  $X_r$  (incoming Kante von  $ext(1)$ ).

3) Zu einem gegebenen Dschungel  $H$  bestimmt man die Aufzählung der Hyperkanten in  $vl$ -Ordnung wie folgt:

- a) Lege eine Arbeitskopie  $K$  von  $H$  an, setze  $A_{vl} = \epsilon$ .
  - b) Falls  $X_r$  markiert ist, Stop!, sonst setze  $X_s = X_r$ .
  - c1) Falls  $X_v$  unmarkiert und Eingangs-Kante von  $nod(X_s, i)$ ,  $i < rank_\Sigma(X_s)$ , ist und die Eingangs-Kanten der Knoten  $nod(X_s, j)$ ,  $j < i$  markiert sind, dann setze  $X_s = X_v$  und gehe zu c1).  
(Bestimmung des „kleinsten“ Vorgängers)
  - c2) Falls  $X_s$  keinen unmarkierten Vorgänger hat, setze  $A_{vl} = A_{vl}X_s$ , markiere  $X_s$  und gehe zu b)
- 4) Ungleich schwieriger wäre es, die Aufzählung umgekehrt aufzubauen.

# Kapitel 2

## Dschungel und Grammatikeigenschaften

In diesem Abschnitt soll untersucht werden, welche weiteren Eigenschaften eine Hypergraph-Grammatik hat, die sich dadurch auszeichnet, daß alle Satzformen Dschungel sind. Ob alle Satzformen Dschungel sind, ist eine Eigenschaft, deren Überprüfung unmittelbar an der Grammatik schwierig ist. Im Gegensatz dazu ist z. B. die Frage, ob der Rang eines jeden Zeichens einer Grammatik größer als 0 ist, unmittelbar entscheidbar.

Es wird sich erfreulicherweise zeigen, daß die Eigenschaft „alle Satzformen sind Dschungel“ äquivalent ist mit einer Reihe von Eigenschaften, die „leichter“ zu bestimmen sind.

Es wird von reduzierten Grammatiken ausgegangen. Es gilt:

**Definition 2.1 (Dschungel-Eigenschaft)** *Eine kontextfreie Hypergraph-Grammatik  $G = (\Sigma, \Delta, P, S)$  besitzt die Dschungel-Eigenschaft genau dann, wenn die folgenden Eigenschaften für  $G$  gelten:*

- a) *Für jede Produktion  $\pi = (X, H) \in P$  gilt:  $H$  ist zyklensfrei.*
- b)  *$\text{rank}_\Sigma(X) > 0$  für alle  $X \in \Sigma$  (keine 0-Hyperkanten).*
- c)  *$\text{rank}_\Sigma(S) = 1$  (1-Hypergraph).*
- d) *Jede Produktion  $\pi = (X, H)$  aus  $G$  hat die folgenden Eigenschaften:*
  1. *alle internen Knoten von  $H$  (also alle neu erzeugten) haben in  $H$  genau eine Eingangs-Kante.*

2. die ersten  $(\text{rank}(H) - 1)$  externen Knoten haben keine Eingangs-Kante in  $H$ .
3.  $\text{ext}(\text{rank}(H))$  hat in  $H$  genau eine Eingangs-Kante.

**Bemerkung:**

Die Eigenschaften a), b) und d) bedeuten, daß auf der rechten Seite einer Produktion jeweils ein  $\text{rank}(X)$ -Dschungel mit  $\text{rank}(X) - 1$  Variablen steht.

Die in obiger Definition vorgenommene Bezeichnung motiviert sich durch den folgenden Satz:

**Satz 2.2** *Alle Satzformen einer Hypergraph-Grammatik  $G$  sind Dschungel  $\Leftrightarrow G$  besitzt Dschungel-Eigenschaft.*

**Beweis:**

„ $\Rightarrow$ “

Sei  $G = (\Sigma, \Delta, P, S)$  eine Grammatik mit der Eigenschaft, daß alle Satzformen Dschungel sind ( $S \Rightarrow^* H$  daraus folgt  $H$  ist Dschungel).

Behauptung: Dann besitzt  $G$  die Dschungel-Eigenschaft.

- a) Alle Satzformen sind zyklensfrei, das Anwenden einer Produktion mit Zyklus auf der rechten Seite würde aber eine Satzform mit Zyklus erzeugen.
- b) Annahme des Gegenteils:  $G$  enthalte ein  $X$  mit  $\text{rank}_\Sigma(X) = 0$ . Da  $G$  reduziert ist, ist  $X$  erreichbar, also existiert eine Satzform  $H$ , die die 0-Hyperkante  $X$  enthält, daher ist  $H$  kein Dschungel  $\Rightarrow$  Widerspruch.
- c) Annahme des Gegenteils:  $\text{rank}_\Sigma(S) \neq 1 \Rightarrow$  der „Start-Hypergraph“ ist kein 1-Hypergraph  $\Rightarrow$  Widerspruch zu  $S$  (Satzform!) ist Dschungel.
- d1) Annahme des Gegenteils:  $G$  enthalte eine Produktion  $\pi = (X, H)$ , deren Hypergraph  $H$  einen internen Knoten  $v$  mit keiner oder mehreren Eingangs-Kanten in  $H$  besitzt.  
Dann gibt es eine Ableitung  $S \Rightarrow^* H' \Rightarrow^\pi H''$ , da  $\pi$  nicht nutzlos ist.  
Die durch Anwendung von  $\pi$  in  $H''$  gegenüber  $H'$  hinzugekommenen Knoten (die internen Knoten von  $H$ ) haben in  $H''$  nur Verbindungen zu Hyperkanten, die durch Anwendung von  $\pi$  erzeugt wurden. Jener Knoten  $v$  ist also in  $H''$  nicht one-incoming und  $H''$  somit kein Dschungel  $\Rightarrow$  Widerspruch zu  $S \Rightarrow^* H''$ .
- d2) Annahme des Gegenteils:  $G$  enthalte eine Produktion  $\pi = (X, H)$ , deren Hypergraph  $H$  einen externen Knoten  $\text{ext}(i), i \in [1 : \text{rank}_\Sigma(X) - 1]$ , mit einer oder mehreren Eingangs-Kanten in  $H$  besitzt.  
Dann gibt es eine Ableitung  $S \Rightarrow^* H' \Rightarrow_e^\pi H''$ , da  $\pi$  nicht nutzlos ist ( $e$  sei die im

letzten Ableitungsschritt ersetzte Hyperkante ( $lab(e) = X$ )).

Da  $H'$  ein Dschungel ist, besitzt der Knoten  $v = nod(e, i)$  genau eine Eingangs-Kante in  $H'$ . Nach Anwendung von  $\pi$  auf  $H'$  besitzt jener Knoten  $v$  dann zwei (oder noch mehr) Eingangs-Kanten,  $H''$  ist also kein Dschungel  $\Rightarrow$  Widerspruch zu  $S \Rightarrow^* H''$ .

(Wegen Zyklenfreiheit darf speziell  $nod(e, i) = nod(e, rank_\Sigma(X))$  in  $H'$  nicht gelten).

- d3) Annahme des Gegenteils:  $G$  enthalte eine Produktion  $\pi = (X, H)$ , deren Hypergraph  $H$  für den externen Knoten  $ext(rank_\Sigma(X))$  keine oder mehrere Eingangs-Kanten in  $H$  besitzt.

Dann gibt es eine Ableitung  $S \Rightarrow^* H' \Rightarrow_e^\pi H''$ , da  $\pi$  nicht nutzlos ist ( $e$  sei die im letzten Ableitungsschritt ersetzte Hyperkante ( $lab(e) = X$ )).

Da  $H'$  ein Dschungel ist, besitzt der Knoten  $v = nod(e, rank_\Sigma(X))$  genau eine Eingangs-Kante in  $H'$ . Offensichtlich ist  $e$  diese Eingangs-Kante. Nach Anwendung von  $\pi$  auf  $H'$  besitzt jener Knoten  $v$  dann keine oder mehrere Eingangs-Kanten,  $H''$  ist also kein Dschungel  $\Rightarrow$  Widerspruch zu  $S \Rightarrow^* H''$ .

Wenn  $G$  also eine Hypergraph-Grammatik ist mit der Eigenschaft, daß alle Satzformen Dschungel sind, dann besitzt  $G$  die Dschungel-Eigenschaft.

„ $\Leftarrow$ “

Sei  $G = (\Sigma, \Delta, P, S)$  eine Grammatik mit Dschungel-Eigenschaft.

Behauptung: Dann ist jede Satzform  $H$  von  $G$  ein Dschungel (ein 1-Dschungel mit 0 Variablen).

Wegen  $S \Rightarrow^* H$ ,  $rank_\Sigma(S) = 1$  ist  $H$  ein 1-Hypergraph.

(a) und

(b) trivial, da 0 Variablen.

(c) Die Knoten aller Satzformen sind one-incoming:

*Induktionsanfang:* Der Starthypergraph ist ein 1-Hypergraph, der einzige Knoten dieses Hypergraphen hat die Hyperkante mit dem Label  $S$  als Eingangs-Kante (wegen c)).

*Induktionsannahme:* Aus  $S \Rightarrow^n H'$  folgt: (die Satzform)  $H'$  ein 1-Hypergraph mit ausschließlich one-incoming Knoten.

*Induktionsschritt:*  $S \Rightarrow^{(n+1)} H$  entspricht  $S \Rightarrow^n H' \Rightarrow H$ ,  $H'$  ist nach Induktionsannahme ein 1-Hypergraph mit ausschließlich one-incoming Knoten.

Um  $H$  aus  $H'$  abzuleiten, muß eine Kante  $e$  mit  $lab(e) = X$  aus  $H'$  gemäß einer Produktion  $\pi = (X, H_x)$  aus  $G$  durch den Hypergraphen  $H_x$  ersetzt werden.

In  $H'$  sind alle Knoten one-incoming. Alle durch Anwendung von  $\pi$  hinzugekommenen Knoten (interne Knoten von  $H_x$ ) sind (wegen d)1.) one-incoming. Die Knoten  $nod(e, i)$ ,  $i \in [1, rank_\Sigma(X) - 1]$ , aus  $H'$  haben in  $H'$  eine Eingangs-Kante und bekommen durch Anwendung von  $\pi$  keine dazu (wegen d)2.). Der

Knoten  $nod(e, rank_{\Sigma}(X))$  hat  $e$  als einzige Eingangs-Kante in  $H'$ , dieser ist auch in  $H$  one-incoming, da  $ext(rank_{\Sigma}(X))$  in  $H_x$  eine Eingangs-Kante hat (wegen d)3.).

Damit sind auch in  $H$  alle Knoten one-incoming.

*Induktionsschluß:*  $S \Rightarrow^n H'$  daraus folgt  $H'$  ist ein 1-Hypergraph mit ausschließlich one-incoming Knoten gilt für alle  $n \in \mathbb{N}$ .

- (d) (zyklenfrei) Annahme des Gegenteils: Es gibt eine Ableitung  $S \Rightarrow^* H$  und  $H$  enthält einen Zyklus. Dann gibt es in dieser Ableitung eine erste Satzform mit Zyklus, d.h.  $S \Rightarrow^* H'' \Rightarrow_e^{\pi} H' \Rightarrow^* H$  und  $H''$  enthält im Gegensatz zu  $H'$  keinen Zyklus. Sei  $lab(e) = X$ ,  $\pi = (X, H_x)$  und  $H_r$  der „Rest“ von  $H''$  (also ohne  $e$ ). Dann bilden offensichtlich  $H_r$  und  $H_x$  zusammen  $H'$ . Für einen geschlossenen Pfad (Zyklus) in  $H'$  gibt es jetzt drei Möglichkeiten:
1. Der Pfad liegt vollständig in  $H_r$ , dann existiert er aber auch in  $H''$ , was einen Widerspruch zur Voraussetzung darstellt.
  2. Der Pfad liegt vollständig in  $H_x$ , das kollidiert mit Bedingung a).
  3. Der Pfad durchläuft sowohl  $H_r$  als auch  $H_x$ . Dann gibt es (mindestens) zwei Übergangspunkte, aus Sicht von  $H_x$  einen Beginn- und Endpunkt des Teilpfades in  $H_x$ . Als Endpunkt kommt wegen d)2. nur  $ext(rank_{\Sigma}(X))$  in Frage, als Beginn einer der übrigen. Daraus folgt dann für den anderen Teil des Pfades in  $H_r$ , daß er bei  $ext(rank_{\Sigma}(X))$  beginnt und entsprechend woanders endet. Diesen anderen Teil gibt es also auch in  $H''$ , und dort schließt dann  $e$  den Zyklus, womit wieder ein Widerspruch zur Voraussetzung vorliegt.
- (e) (keine 0-Hyperkanten) folgt aus der Bedingung b).

Wenn also  $G$  eine Hypergraph-Grammatik mit Dschungel-Eigenschaft ist, dann sind alle Satzformen von  $G$  Dschungel. □

Mit Hilfe dieses Satzes ist es jetzt möglich, die Frage, ob alle Satzformen einer gegebenen Hypergraph-Grammatik Dschungel sind, durch Überprüfen der Dschungel-Eigenschaften zu beantworten.

## 2.1 Reiner Dschungel und Grammatikeigenschaften

In diesem Abschnitt soll gezeigt werden, daß gewisse Eigenschaften einer kontext-freien Hypergraph-Grammatik  $G$  dazu führen, daß alle Satzformen reine Dschungel sind. Diese Eigenschaften stellen natürlich eine „Verschärfung“ der Dschungel-Eigenschaft dar.

Umgekehrt wird gezeigt, daß es zu jeder kontextfreien Hypergraph-Grammatik  $G'$ , deren Satzformen alle Dschungel sind, eine kontextfreie Hypergraph-Grammatik  $G$  mit  $L(G) = L(G')$  existiert, die oben angedeutete Eigenschaften besitzt.

**Definition 2.3 (reiner-Dschungel-Eigenschaft)** *Eine kontextfreie Hypergraph-Grammatik  $G = (\Sigma, \Delta, P, S)$  besitzt die reiner-Dschungel-Eigenschaft, wenn sie folgende Bedingungen erfüllt:*

- a) *Für jede Produktion  $\pi = (X, H) \in P$  gilt:  $H$  ist zyklensfrei.*
- b)  *$\text{rank}_\Sigma(X) > 0$  für alle  $X \in \Sigma$  (keine 0-Hyperkanten).*
- c)  *$\text{rank}_\Sigma(S) = 1$  (1-Hypergraph).*
- d) *Jede Produktion  $\pi = (X, H)$  aus  $G$  hat die folgenden Eigenschaften:*
  - 1. *alle internen Knoten von  $H$  (also alle neu erzeugten) haben in  $H$  genau eine Eingangs-Kante.*
  - 2. *die ersten  $(\text{rank}(H) - 1)$  externen Knoten haben keine Eingangs-Kante in  $H$ .*
  - 3.  *$\text{ext}(\text{rank}(H))$  hat in  $H$  genau eine Eingangs-Kante.*
- e) *Für jede Produktion  $\pi = (X, H)$  gilt:*
  - 1. *Von jeder Hyperkante von  $H$  führt ein Pfad zu  $\text{ext}(\text{rank}(H))$ .*
  - 2. *Von jedem Knoten  $\text{ext}(i)$ ,  $i < \text{rank}(H)$  führt ein Pfad zu  $\text{ext}(\text{rank}(H))$ .*

Mit dem folgenden Satz zeigt sich der Sinn der in Definition 2.3 vorgenommenen Benennung:

**Satz 2.4** *Sei  $G = (\Sigma, \Delta, P, S)$  eine kontextfreie Hypergraph-Grammatik mit reiner-Dschungel-Eigenschaft.*

*Dann sind alle Satzformen von  $G$  reine Dschungel.*

**Beweis:**

Die Bedingungen a)-d) der Definition führen dazu, daß jede Satzform ein Dschungel ist (Satz 2.2). Ein Dschungel ist ein reiner Dschungel, wenn von jeder Hyperkante (mindestens) ein Pfad zum externen Knoten des Dschungel führt. Bleibt also zu zeigen, daß die Bedingung e) gewährleistet, daß in jeder Satzform diese Pfade existieren.

Induktion über Länge der Ableitung.

$i = 0$ :

Im Starthypergraphen existiert nur die Hyperkante mit dem Label  $S$ , und von dort

führt ein Pfad zum externen Knoten.

$i \rightarrow i + 1$ :

Sei die Existenz der Pfade für alle Satzformen, die in  $i$  Ableitungsschritten erreicht werden können, bewiesen.

Sei  $H_b = (V_b, E_b, nod_b, lab_b, ext_b)$  eine Satzform mit  $S \Rightarrow^{i+1} H_b$ . Dann gibt es eine Satzform  $H_a = (V_a, E_a, nod_a, lab_a, ext_a)$  mit der Hyperkante  $e$ ,  $lab(e) = X$  und eine Produktion  $\pi = (X, H)$ ,  $H = (V, E, nod, lab, ext)$ , so daß  $S \Rightarrow^i H_a \Rightarrow_e^\pi H_b$  gilt.

Nach Induktionsvoraussetzung gilt für alle Hyperkanten von  $H_a$ , daß ein Pfad zum externen Knoten von  $H_a$  existiert. Die Hyperkanten von  $H_b$  lassen sich in zwei Gruppen aufteilen; jenen, die schon in  $H_a$  vorhanden waren, und jenen, die durch das Einsetzen von  $H$  an der Stelle von  $e$  hinzugekommen sind.

Für erstere müssen zwei Fälle betrachtet werden. Sei  $e_a$  aus dieser Gruppe:

1. In  $H_a$  existiert ein Pfad von  $e_a$  zum externen Knoten, der nicht über  $e$  führt. Dann existiert dieser Pfad auch in  $H_b$ .
2. In  $H_a$  existiert ein Pfad von  $e_a$  zum externen Knoten, der über  $e$  führt. Laufe dieser Pfad über die Knoten  $nod(e, i)$  ( $i < rank_\Sigma(lab(e))$ ) und  $nod(e, rank_\Sigma(lab(e)))$  durch  $e$ .

Dann gibt es in  $H_b$  einen Pfad von  $e_a$ , der quasi über den Knoten  $ext(i)$  von  $H$  (der hier in dieser Art und Weise nicht mehr anzusprechen ist) in  $H$  eintritt, von dort wegen der Bedingung e)2. bis zu  $ext(rank_\Sigma(lab(e)))$  geführt werden kann, und dann wie in  $H_a$  bis zum externen Knoten von  $H_b$  fortgesetzt werden kann.

Formal: In  $H_a$  existiert ein Pfad

$$(e_a = e_0, 0, \dots, v_k, i_k, e_{k+1} = e, 0, v_{k+1}, \dots, 0, v_n = ext_a(1)).$$

Wegen e)2) existiert in  $H$  ein Pfad

$$(ext(i_k) = v'_0, j_0, \dots, 0, v'_m = ext(rank_\Sigma(H))).$$

Also existiert in  $H_b$  der Pfad

$$(e_a = e_0, 0, \dots, v_k = v'_0, j_0, \dots, 0, v'_m = v_{k+1}, \dots, v_n = ext_a(1)).$$

Für die zweite Gruppe, die Hyperkanten, die durch Einsetzen von  $H$  hinzugekommen sind, gilt:

Sei  $e_{neu}$  eine Hyperkante dieser Gruppe:

Wegen Bedingung e)1. führt ein Pfad zu jenem Knoten, der in  $H_a$  mit  $nod(e, rank_\Sigma(lab(e)))$  anzusprechen war. Da in  $H_a$  ein Pfad von  $e$  zum externen Knoten von  $H_a$  existierte, kann jeder Pfad von dort aus in  $H_b$  zum externen Knoten fortgesetzt werden.

Formal: In  $H_a$  existiert ein Pfad

$$(e = e_0, 0, v_0, \dots, 0, v_n = ext_a(1)),$$

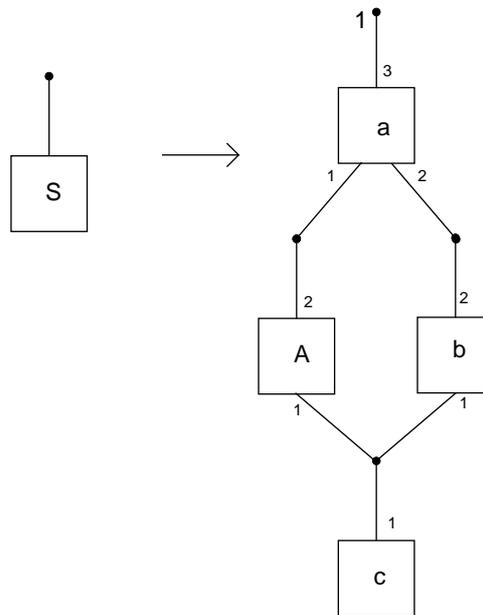
Wegen e)1) existiert in  $H$  ein Pfad

$$(e_{neu} = e'_0, 0, \dots, 0, v'_m = ext(rank_\Sigma(H))).$$

Also existiert in  $H_b$  der Pfad

$$(e_{neu} = e'_0, 0, \dots, 0, v'_m = v_0, \dots, 0, v_n = ext_b(1)).$$

*Induktionsschluß:* Gilt für alle  $i$ .

Abbildung 2.1: Produktion  $p_1$ 

Also ist jede Satzform von  $G$  ein reiner Dschungel.

□

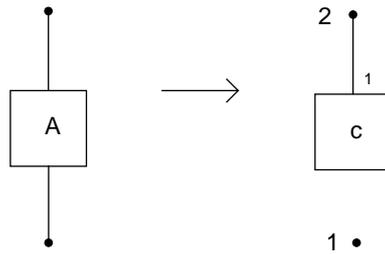
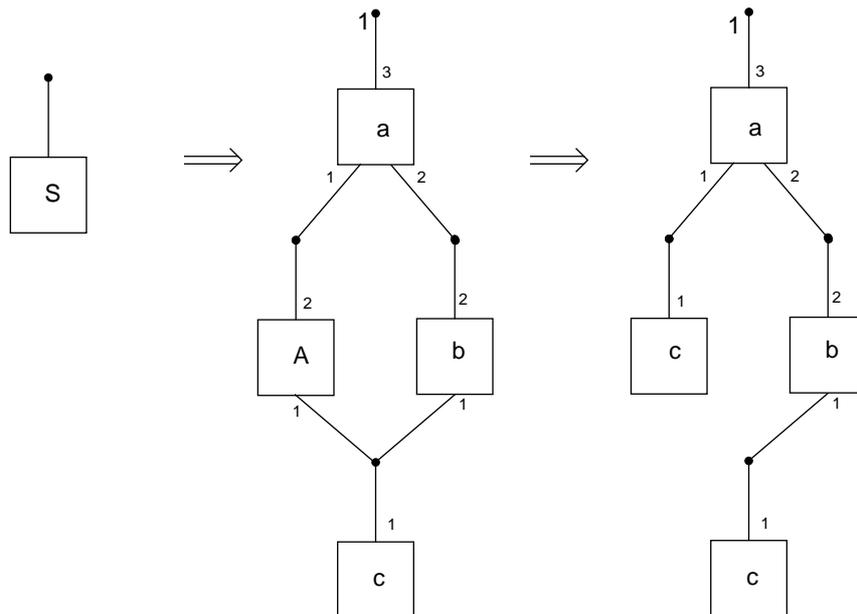
Im Falle der Dschungel-Eigenschaft bestand eine Äquivalenz zwischen gewissen Grammatikeigenschaften einerseits und der Eigenschaft, daß alle Satzformen Dschungel sind, andererseits. Dies ist bei der reinen-Dschungel-Eigenschaft nicht der Fall.

Betrachten wir dazu das folgende Beispiel:

**Beispiel 2.5** Sei  $G = (\{S, A, a, b, c\}, \{a, b, c\}, \{p_1, p_2\}, S)$  eine kontextfreie Hypergraph-Grammatik mit  $p_1$  wie in Abbildung 2.1 und  $p_2$  wie in Abbildung 2.2. Dann ist nur eine einzige Ableitung wie in Abbildung 2.3 möglich. Leicht erkennt man, daß alle (drei) Satzformen reine Dschungel sind, aber die Produktion  $p_2$  erfüllt die Bedingung e2) der Definition 2.3 nicht, da vom Knoten  $ext(1)$  kein Pfad zum Knoten  $ext(2)$  führt.

Wir zeigen nun, daß es zu einer gegebenen kontextfreien Hypergraph-Grammatik  $G$  eine Hypergraph-Grammatik  $G'$  mit reiner-Dschungel-Eigenschaft gibt, so daß  $L(G) = L(G')$  gilt:

**Satz 2.6** Sei  $G = (\Sigma, \Delta, P, S)$  eine kontextfreie Hypergraph-Grammatik mit der Eigenschaft, daß alle Satzformen von  $G$  reine Dschungel sind. Dann gibt es ei-

Abbildung 2.2: Produktion  $p_2$ Abbildung 2.3: Ableitung in  $G$

ne kontextfreie Hypergraph-Grammatik  $G'$  mit reiner-Dschungel-Eigenschaft, so daß  $L(G) = L(G')$  gilt.

**Beweis:**

- a) Alle Satzformen von  $G$  sind reine Dschungel, daraus folgt, daß alle Satzformen Dschungel sind und damit (nach Satz 2.2) die Dschungel-Eigenschaft für  $G$ . Somit erfüllt  $G$  die Bedingungen a)-d).
- b) Betrachten wir die Bedingung e)1. und nehmen an, daß die Produktion  $p = (X, H) \in P$  diese nicht erfülle, weil von der Hyperkante  $e$  kein Pfad zu  $ext(rank(H))$  führt. Sei  $S \Rightarrow^* H_a \Rightarrow^p H_b$  eine Ableitung in  $G$ , dann führt in der Satzform  $H_b$  von der Hyperkante  $e$  kein Pfad zum externen Knoten von  $H_b$ . Also wäre  $H_b$  kein reiner Dschungel, was einen Widerspruch darstellt.
- c) Bleibt die Bedingung e)2.:  
Falls  $G$  diese Bedingung erfüllt, setzen wir  $G' = G$  und sind fertig.  
Der andere Fall ist der, daß in einer Produktion  $p = (X, H) \in P$ ,  $H = (V, E, nod, lab, ext)$  von den Knoten  $ext(i_1), \dots, ext(i_j)$ ,  $j \geq 1, i_k < rank(H)$ ,  $k \in [1 : j]$ , kein Pfad zum Knoten  $ext(rank(H))$  führt. Wäre eine Kante  $e$  Ausgangskante für einen solchen Knoten, dann würde wegen e)1. aber ein Pfad existieren. Also sind diese Knoten isoliert (d. h. :  $\nexists e \in E, l \in \mathbb{N} : nod(e, l) = ext(i_k)$ ). Konstruiere die Grammatik  $G_{help} = (\Sigma_h, \Delta, P_h, S_h)$  wie folgt:
  - 0)  $p_{neu} = (X_{neu}, H_{neu})$  mit  $rank_{\Sigma_h}(X_{neu}) = rank(H) - j$ ,  $H_{neu}$  ist  $H$  ohne die Knoten  $ext(i_1), \dots, ext(i_j)$ .
  - 1)  $\Sigma_h = \Sigma \cup \{X_{neu}\}$ .
  - 2a)  $P_h = \{q' = (X_q, H'_q) \mid \exists q = (X_q, H_q) \in (P \setminus \{p\}) \cup \{p_{neu}\} : H'_q \text{ ergibt sich aus } H_q, \text{ indem einige (0 bis alle) mit } X \text{ gelabelten Hyperkanten durch mit } X_{neu} \text{ gelabelte ersetzt werden. Bei der Ersetzung müssen entsprechend die Tentakel } i_1, \dots, i_j \text{ gekappt werden}\}$ .
  - 3)  $S_h = S$  (wegen  $rank_{\Sigma}(S) = 1$  folgt  $X \neq S$ ).
  - 4) Reduziere  $G_{help}$ : Falls  $X$  nicht mehr auf der linken Seite einer Produktion auftaucht, streiche  $X$  aus  $\Sigma_h$  und alle Produktionen aus  $P_h$ , bei denen  $X$  auf der rechten Seite auftritt.

Im wesentlichen ist die Produktion  $p$  aus  $G$  durch die Produktion  $p_{neu}$  in  $G_{help}$  ersetzt worden. Es ist also einzusehen, daß  $L(G_{help}) = L(G)$  gilt.

Weitergehend gilt sogar, sei  $H_G$  eine Satzform von  $G$  ( $S \Rightarrow^* H_G$ ), dann gibt es eine (Menge von) Satzform(en) in  $G_{help}$ , die sich von  $H_G$  nur dadurch unterscheidet, daß an Stellen, wo in  $H_G$  eine Hyperkante mit dem Label  $X$  steht, dort eine mit dem Label  $X_{neu}$  steht (mit entsprechend verringerter Anzahl von

Tentakeln).

Ebenso besteht eine Ähnlichkeit zwischen den Ableitungen dorthin: Sei  $p_1, \dots, p_n$  eine Folge von Ableitungen, die von  $S$  zu  $H_G$  führt. Dann gibt es in  $G_{help}$  eine Folge  $p'_1, \dots, p'_n$  von Ableitungen (angewendet auf die „entsprechenden“ Hyperkanten), die zu einem „Bild“ von  $H_G$  führt, mit:

$$p'_i = \begin{cases} p_{neu} & \text{falls } p_i = p \wedge p \notin P_X \\ p'_{neu} \in P' & \text{falls } p_i = p \wedge p \in P_X \\ q' \in P' & \text{falls } p_i = q \in P_X \setminus \{p_{neu}\} \\ p_i & \text{sonst} \end{cases}$$

Außerdem besitzt  $G_{help}$  offenbar Dschungel-Eigenschaft und erfüllt die Bedingung e)1..

Das obige Verfahren (unter (c)) wiederhole man für  $G_{help}$  solange, bis das Ergebnis die Bedingung e)2. erfüllt. Dann setze man  $G' = G_{help}$ . Offenbar gilt  $L(G') = L(G)$ , es ist also konstruktiv eine kontextfreie Hypergraph-Grammatik mit reiner-Dschungel-Eigenschaft gefunden worden.

□

**Bemerkung:**

Das beschriebene Verfahren ist ein abbrechendes, die Argumentation ist ähnlich zu jener, die man anwendet, wenn es um die Beseitigung von  $\epsilon$ -Produktionen in kontextfreien Grammatiken geht.

Kehren wir noch einmal zu Beispiel 2.5 zurück und wenden darauf das im Beweis gegebene Verfahren an. Dann erhalten wir:

**Beispiel 2.7**  $G' = (\{S, A_{neu}, a, b, c\}, \{a, b, c\}, \{p'_1, p'_2\}, S)$  ist eine kontextfreie Hypergraph-Grammatik mit  $p'_1$  wie in Abbildung 2.4 und  $p'_2$  wie in Abbildung 2.5. Die einzige Ableitung sieht wie in Abbildung 2.6 aus. Offensichtlich sind die terminalen Hypergraphen der Ableitungen in Abbildung 2.3 und Abbildung 2.6 identisch.

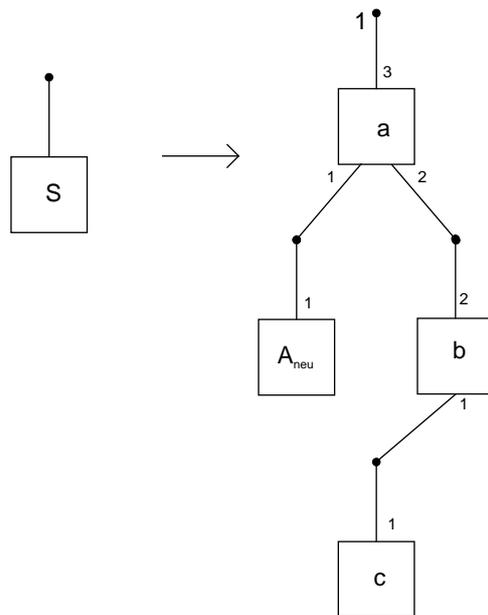


Abbildung 2.4: Produktion  $p'_1$

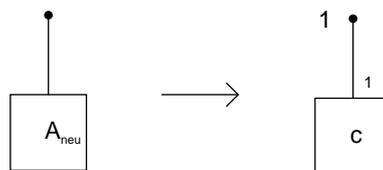
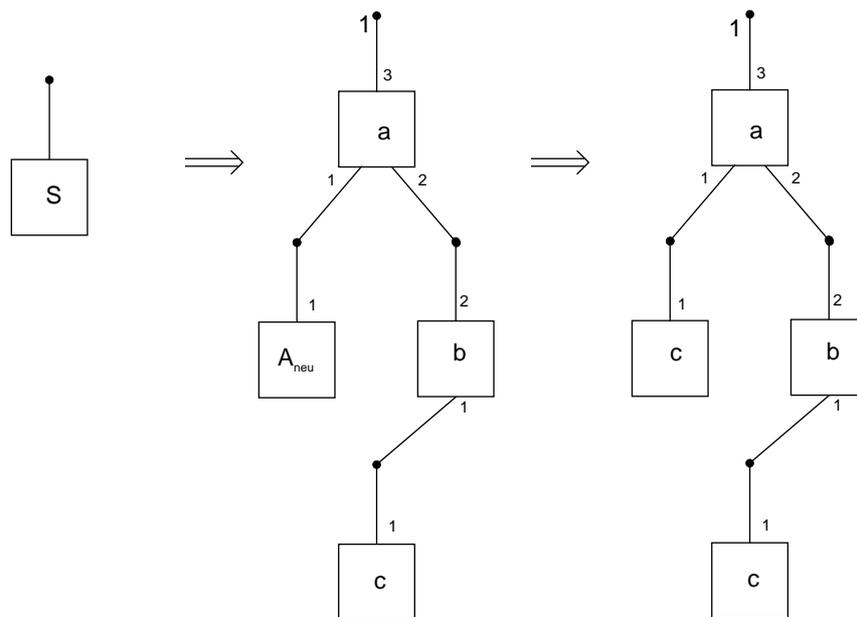


Abbildung 2.5: Produktion  $p'_2$

Abbildung 2.6: Ableitung in  $G'$

# Kapitel 3

## Der Iterationssatz im Dschungel

A. Habel hat in [Hab92] einen Iterationssatz für kontextfreie Hypergraph-Grammatiken vorgestellt.

Wegen seiner Bedeutung auch für diese Arbeit geben wir zunächst skizzenförmig die wesentlichen Überlegungen an, die zu dem Satz führen. Den Satz selbst geben wir für den Spezialfall der Dschungel an, von einem Zitat sehen wir ab – ansonsten müßten eine Reihe von neuen Begriffen eingeführt werden, da Frau Habel eine andere Terminologie verwendet. Es wird untersucht, welche weitergehenden Feststellungen gemacht werden können, wenn die Dschungel-Eigenschaft für die Hypergraph-Grammatik vorausgesetzt wird.

### 3.1 Der Iterationssatz

Beim Iterationssatz für kontextfreie Grammatiken – dessen Kenntnis und Beweisidee hier als bekannt vorausgesetzt wird – beginnen die Betrachtungen beim Ableitungsbaum. Ist dieser genügend groß, so existiert mindestens ein Pfad von der Wurzel zu einem Blatt, der so lang ist, daß dort nichtterminale Symbole – als rechte Seitenknotenmarkierender Produktionen – wiederholt auftreten müssen.

Ganz entsprechend verhält es sich bei kontextfreien Hypergraph-Grammatiken – weswegen wir bei unseren Erläuterungen oft die Parallele heranziehen werden.

Betrachten wir die Skizze eines Ableitungsbaums aus Abbildung 3.1 .

Es seien  $u$ ,  $v$ ,  $x$ ,  $y$  und  $z$  die durch den jeweiligen Teil des Ableitungsbaums festgelegten *terminalen* Teilworte bzw. Teilhypergraphen.

Im Falle der kontextfreien Grammatik können die Ableitungen

$$S \Rightarrow^k uAz \Rightarrow^l uvAyz \Rightarrow^m uvxyz = w$$

„abgelesen“ werden, weswegen z. B. auch

$$uxz (S \Rightarrow^k uAz \Rightarrow^m uxz)$$

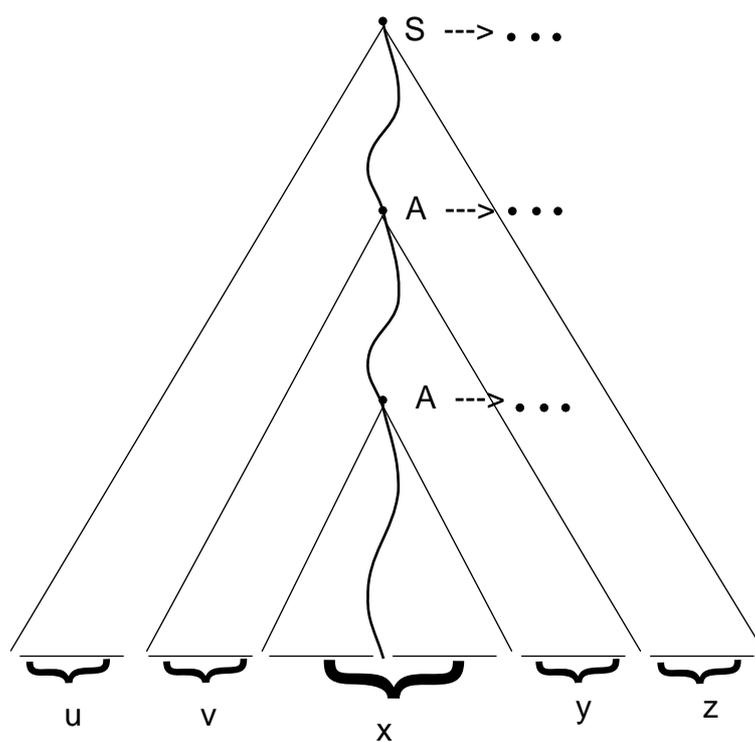


Abbildung 3.1: Skizze eines Ableitungsbaums

als zur Sprache gehörig geschlossen werden kann. In den Abbildungen 3.2, 3.3 und 3.4 sind die Entsprechungen für die kontextfreie Hypergraph-Grammatik skizziert.

Dabei sind alle Hyperkanten aus  $H_1 - A$ ,  $H_2 - A$  und  $H_3$  mit terminalen Symbolen beschriftet.

$H_1 - A$  entspricht „ $uz$ “ und neben dem externen Knoten sind (maximal)  $n = \text{rank}(A)$  – sozusagen interne externe – Knoten besonders zu beachten. Diese Knoten bezeichnen wir als link-Knoten.

$H_2 - A$  entspricht „ $vy$ “ und hat in der Abbildung 3.3 jeweils  $n$  äußere und innere link-Knoten.  $H_3$  schließlich entspricht „ $x$ “.

Wir stellen die Hypergraphen  $H_1$ ,  $H_2$  und  $H_3$  noch einmal getrennt und graphisch „günstiger“ in Abbildung 3.5 dar.

Man kann zunächst  $H_1$  aus  $S$  ableiten, dann aus dem  $A$  einmal oder mehrmals oder auch keinmal  $H_2$  ableiten und schließlich  $H_3$ .

Bei A. Habel heißen die entsprechenden Teilhypergraphen – aus deswegen naheliegenden Gründen – *first*, *link* und *last* und mit diesen Bezeichnungen sieht die Visualisierung des Iterationssatzes wie in Abbildung 3.6 aus. A. Habel hat die Teilgraphen nebeneinander gezeichnet, da wir aber meist den externen Knoten des Dschungels oben zeichnen, ist für unsere Zwecke die hier gewählte Darstellung passender.

Bevor wir den Satz angeben, müssen wir zunächst das „Zusammenfügen“ der Hypergraphen mittels der link-Knoten beschreiben:

**Definition 3.1 (linken)** Seien  $H_1 = (V_1, E_1, \text{nod}_1, \text{lab}_1, \text{ext}_1)$  und  $H_2 = (V_2, E_2, \text{nod}_2, \text{lab}_2, \text{ext}_2)$  zwei Hypergraphen mit  $V_1 \cap V_2 = E_1 \cap E_2 = \emptyset$ .

Seien  $\text{link}_1 = v_1^1 \dots v_n^1 \in V_1^n$  und  $\text{link}_2 = v_1^2 \dots v_n^2 \in V_2^n$  ( $n \geq 1$ ) zwei Aufzählungen von je  $n$  link-Knoten. Dabei besteht  $\text{link}_2$  aus  $n$  paarweise disjunkten Knoten.

Dann ist  $H_1 \otimes_{(\text{link}_1, \text{link}_2)} H_2 = (V, E, \text{nod}, \text{lab}, \text{ext})$  mit

- $V = V_1 \cup (V_2 - \{v \mid \exists w, x \in V_2^* : \text{link}_2 = wvx\})$

- $E = E_1 \cup E_2$

- 

$$\text{nod}(e, i) = \begin{cases} \text{nod}_1(e, i) & \text{falls } e \in E_1 \\ v & \text{falls } e \in E_2, \text{nod}_2(e, i) = v_j^2, v = v_j^1 \\ \text{nod}_2(e, i) & \text{sonst (Knoten aus } E_2, \text{ aber nicht im „link“)} \end{cases}$$

- 

$$\text{lab}(e) = \begin{cases} \text{lab}_1(e) & \text{falls } e \in E_1 \\ \text{lab}_2(e) & \text{sonst, also falls } e \in E_2 \end{cases}$$

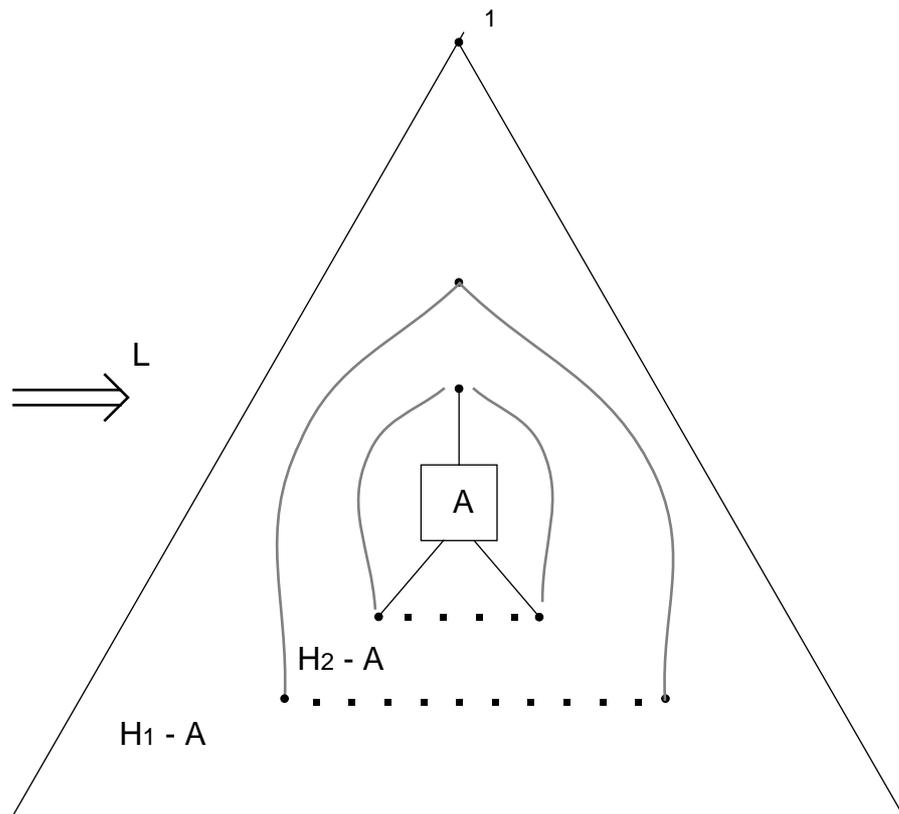
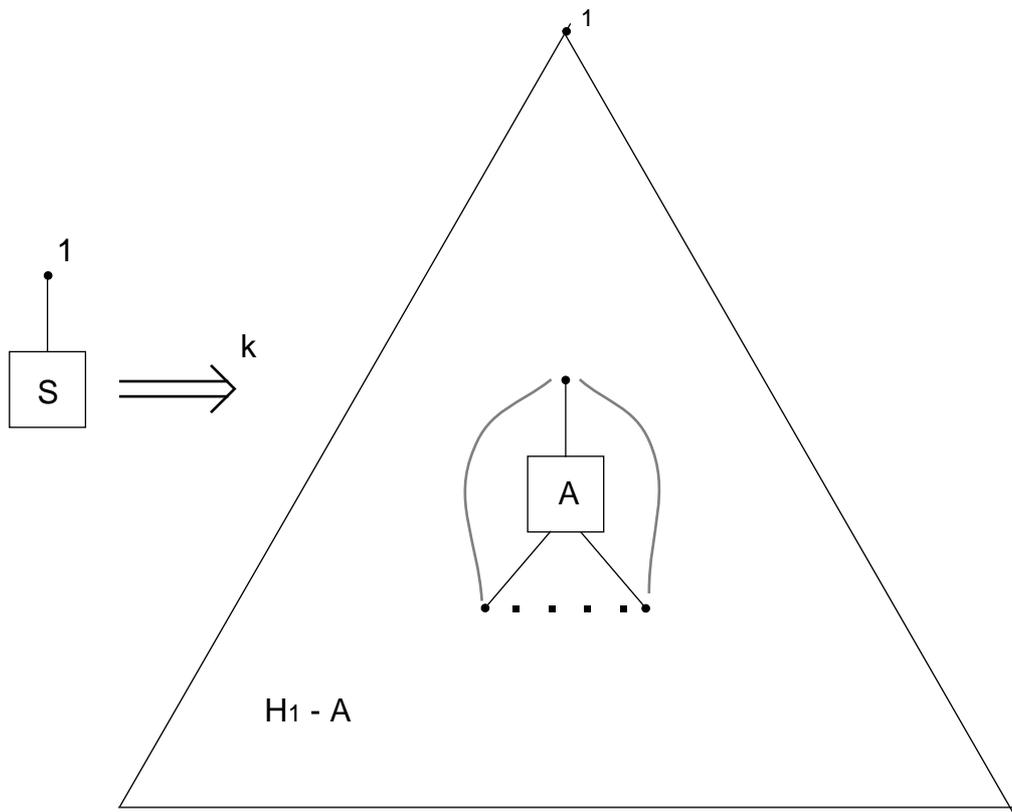


Abbildung 3.2: Ableitung von „ $S \Rightarrow^k uAz \Rightarrow^L uvAyz$ “

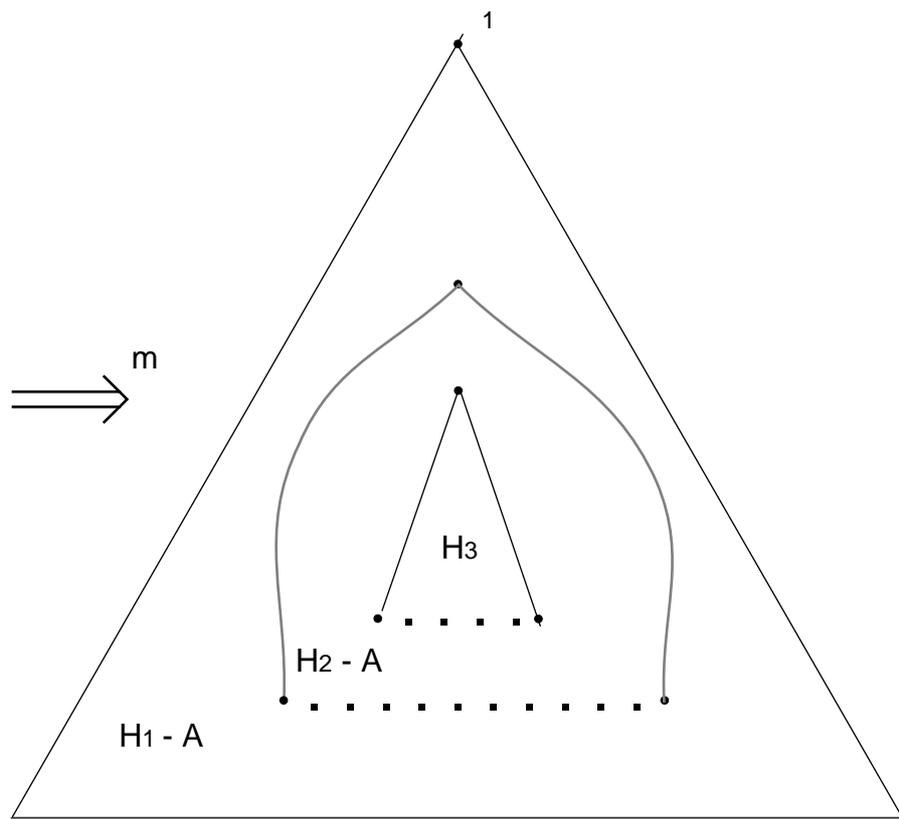
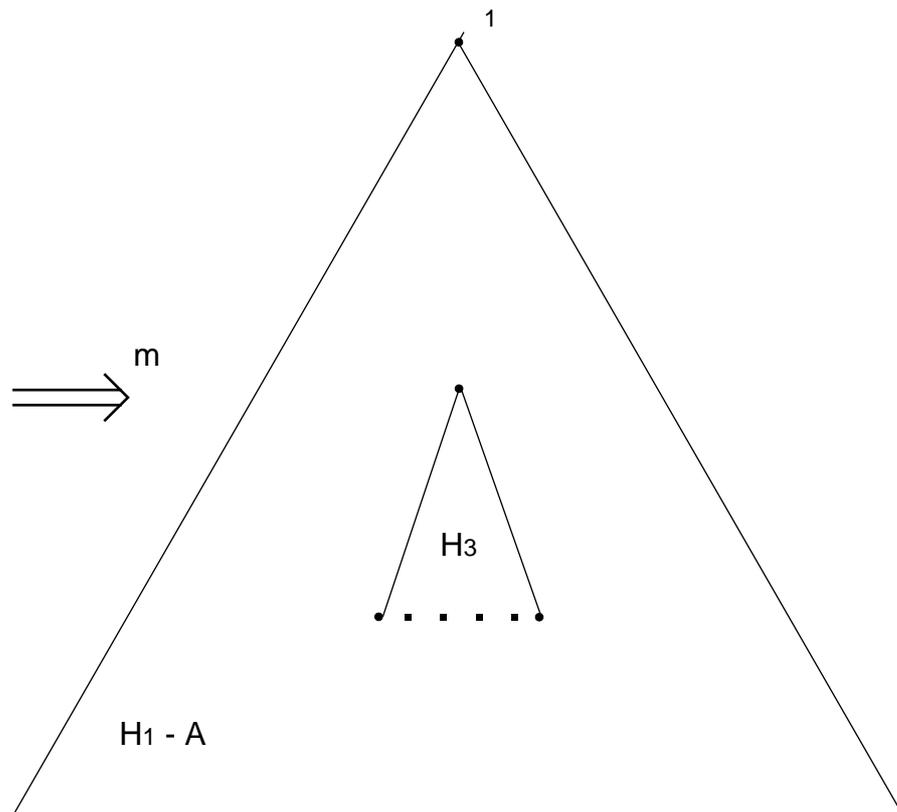
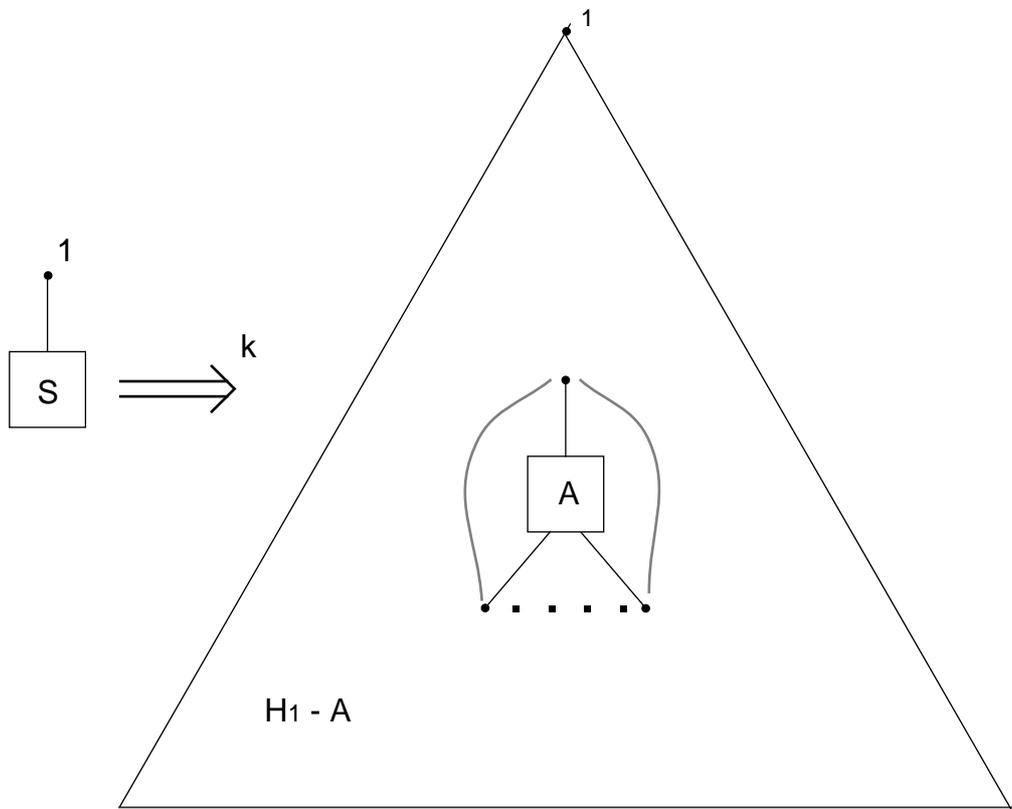


Abbildung 3.3: Ableitung von „ $uvAyz \Rightarrow^m uvxyz$ “

Abbildung 3.4: Ableitung von „ $S \Rightarrow^k uAz \Rightarrow^m uxz$ “

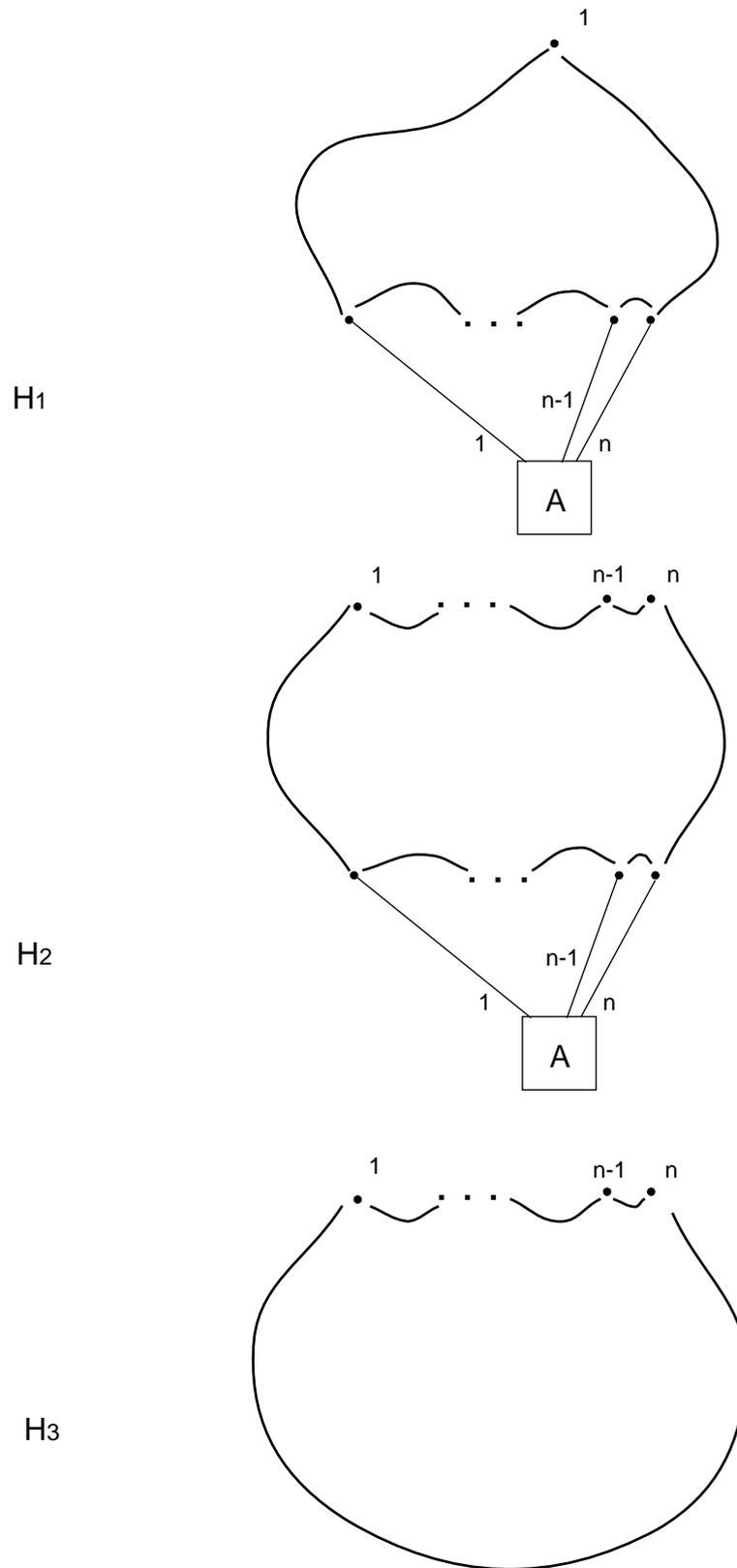


Abbildung 3.5: Andere Darstellung der Hypergraphen  $H_1$ ,  $H_2$  und  $H_3$  aus Abbildung 3.2 und 3.3



- Sei  $ext_2 = y_1 \dots y_m$ , dann sei  $ext'_2 = y'_1 \dots y'_m$  mit

$$y'_i = \begin{cases} v_j^1 & \text{falls } y_i = v_j^2 \\ y_i & \text{sonst} \end{cases}$$

Dann ist  $ext = ext'_2 ext_1$

### Schreibweisen:

Wir schreiben kurz  $H_1 \otimes H_2$ , wenn die link-Knoten aus dem Zusammenhang geschlossen werden können.

Für  $H_1 \otimes (H \otimes (H \otimes (H \otimes H_2)))$  schreiben wir auch  $H_1 \otimes H^3 \otimes H_2$  (hierbei sollte  $H$  sinnvollerweise zwei Aufzählungen von link-Knoten besitzen – je eine für rechts und für links). Daß eine Kette von link-Anweisungen von rechts nach links auszuwerten ist, liegt daran, daß nur für  $link_2$  paarweise disjunkte Knoten gefordert wurden.

### Bemerkungen:

1. Die Forderung nach paarweise disjunkten Knoten in  $link_2$  dient der Wohldefiniertheit. Ansonsten ließe sich eine Situation konstruieren, in der  $nod(e, i) = v_1$  und  $nod(e, i) = v_2$  mit  $v_1 \neq v_2$  für den zusammengefügteten Hypergraphen gilt.
2. Definiert man „großzügiger“ – fordert keine Disjunktheit für  $link_2$  und „klebt“ einfach alles übereinander – gibt es unter Umständen Schwierigkeiten mit den Dschungeleigenschaften (insbesondere: genau eine Eingangskante pro Knoten); und in unserem Fall können wir mit den Einschränkungen gut leben.
3. Das „linken“ ist keine kommutative Verknüpfung. Unter gewissen Bedingungen, z. B. wenn in keinem der beiden  $links$  ein Knoten wiederholt auftritt, liefert es allerdings auch nach Vertauschen der Argumente das gleiche Ergebnis.

**Satz 3.2 (Iterationssatz)** Sei  $G = (\Sigma, \Delta, P, S)$  eine Hypergraph-Grammatik.

Dann existieren Konstanten  $p$ ,  $q$  und  $r$ , die nur von  $G$  abhängen. Ist  $H$  ein Hypergraph aus  $L(G)$  mit  $|H| \geq p$ , dann existieren die Hypergraphen  $first$ ,  $link$  und  $last$  und die link-Knoten  $link_{first}$ ,  $link_{link}^o$ ,  $link_{link}^u$  und  $link_{last}$  so, daß

1.  $first \otimes_{(link_{first}, link_{link}^o)} (link \otimes_{(link_{link}^u, link_{last})} last) = H$
2.  $|link| + |last| \leq q$
3.  $|link| \geq 1$  ( $link$  ist nicht trivial)
4.  $|link_{first}| = |link_{link}^o| = |link_{link}^u| = |link_{last}| \leq r$
5. sei  $i \in \mathbb{N}_0$ , dann ist  $first \otimes link^i \otimes last \in L(G)$

**Bemerkung:**

Die Zahlen können genauer angegeben werden.

Sei  $n = |\Sigma - \Delta|$ ,  $m = \max\{|H'| \mid p = (B, H') \in P\}$  und  $k = \max\{\text{rank}(x) \mid x \in \Sigma\}$ .

Dann ist  $p = m^{n+1} + 1$ ,

$q = m^{n+1}$  und

$r = k$ .

Betrachten wir hierzu ein kleines Beispiel:

**Beispiel 3.3** Sei  $G = (\{1, f, g, h, A, S\}, \{1, f, g, h\}, P, S)$  eine Hypergraph-Grammatik mit den Produktionen aus Abbildung 3.7.

Es ist  $n = 2$  und  $m = 3$ .

In  $L(G)$  ist z. B. „ $h(f^3(1), g^3(1))$ “ (siehe Abb. 3.8). In Abb. 3.9 ist die „Zerlegung“ von „ $h(f(1), g(1))$ “ angegeben. Man überzeuge sich, daß diese Zerlegung das im Satz 3.2 gewünschte leistet. Zur Erinnerung: Man kann diese Zerlegung „konstruieren“, indem man den Anfang der Ableitung betrachtet (bei dieser einfachen Hypergraph-Grammatik sehen wir von einer Betrachtung des Ableitungsbaums ab) - siehe Abbildung 3.10:

Nach dem ersten Ableitungsschritt taucht das erste Mal das nichtterminale Symbol  $A$  auf, alles, was bis hierher an terminalen Symbolen produziert wurde, gehört zu *first* (- in diesem Fall eine Hyperkante mit Label 1 -).

Nach dem zweiten Ableitungsschritt taucht das nichtterminale Symbol  $A$  zum zweiten Mal auf, alles, was „inzwischen“ produziert wurde, gehört zu *link* (- in diesem Fall eine Hyperkante mit Label  $g$  und eine mit  $f$  -).

Da  $\text{rank}(A) = 3$  ist, müssen 3 *link*-Knoten vorhanden sein, Diese können anhand der Tentakel von  $A$  in den jeweiligen Satzformen bestimmt werden.

Zwei Dinge wirken etwas irritierend im obigen Beispiel. Zum einen der Knoten  $v_1$ , der im  $link_{first}$  zweimal auftritt, und zum anderen der Knoten  $v_5$ , der sowohl in  $link_{link}^o$  als auch in  $link_{link}^u$  auftaucht. Sie erschweren es dem Betrachter, die Teilhypergraphen vor dem geistigen Auge zusammzusetzen.

Deswegen werden wir in weiteren Zeichnungen mit „Knotenkopien“ arbeiten, die über eine (gerichtete) „Identitätskante“ mit dem Original verbunden sind.

Die drei Teilgraphen aus Beispiel 3.3 (Abbildung 3.9) sehen dann wie in Abbildung 3.11 aus.

Wir können an dieser Abbildung noch ein weiteres Prinzip festmachen:

Die *link*-Knoten werden jeweils von links nach rechts auf einer Höhe gezeichnet, die ihrer Funktion „entspricht“ – z.B. die oberen *link*-Knoten von *link* ganz oben in *link*.

Damit dieses Prinzip nicht mit jenem kollidiert, den höchsten externen Knoten eines Dschungels ganz oben einzuzichnen, haben wir auch eine „Kopie“ von  $v_2$  angelegt. Außerdem sei bemerkt, daß die Richtung der Identitätskante z. B. zwischen  $v_5$  und

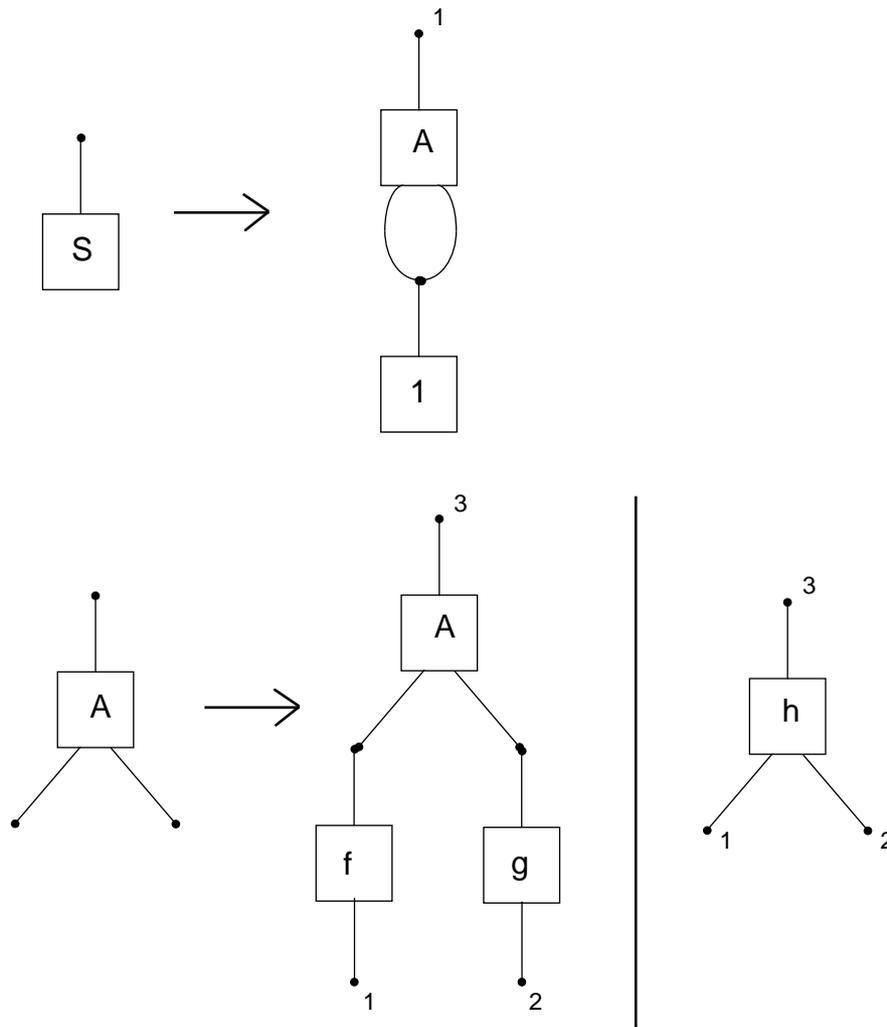
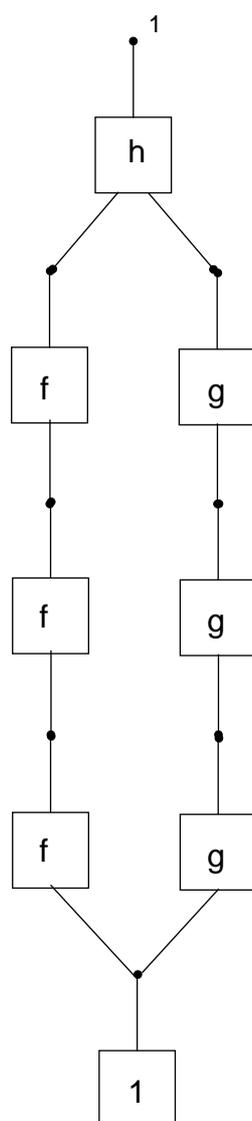


Abbildung 3.7: Produktionen von  $G$

Abbildung 3.8: „ $h(f^3(1), g^3(1))$ “

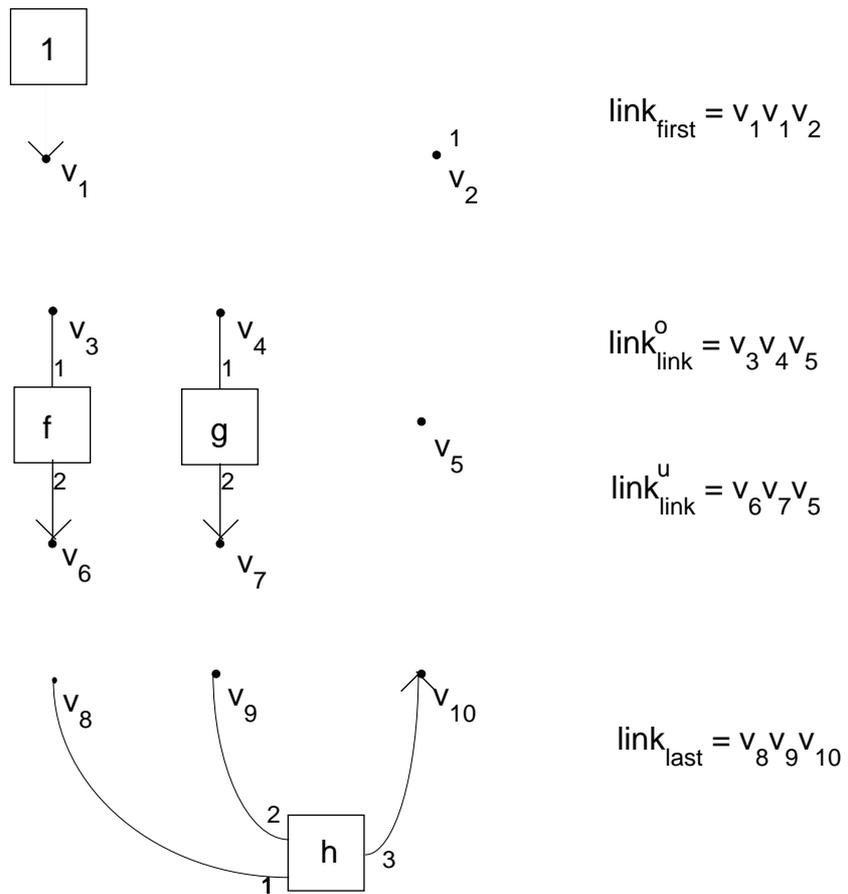


Abbildung 3.9: Zerlegung von „ $h(f(1), g(1))$ “

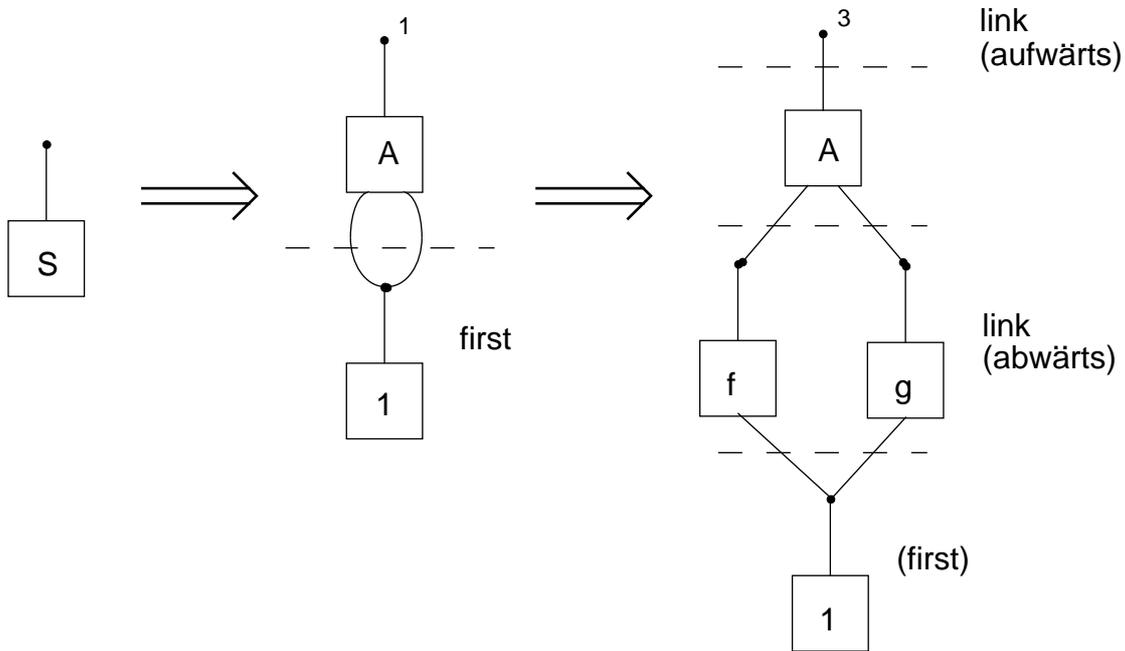


Abbildung 3.10: Beginn der Ableitung

$v_5$  dadurch motiviert ist, daß der untere  $v_5$  Knoten mit  $v_{10}$  beim linken „vereint“ wird und  $v_{10}$  eine Eingangskante besitzt – was bei Dschungeln die Folgerung zuläßt, daß in diesem Fall  $v_2$  keine Eingangskante (vor dem linken) besitzt.

Wenn wir jetzt die speziellen Eigenschaften der Hypergraphen *first*, *link* und *last* bei Hypergraph-Grammatiken mit Dschungel-Eigenschaft untersuchen, müssen wir folgende Unterscheidung beachten: Solche Eigenschaften, die für alle diese *first*, *link* und *last* gelten, und solche, die darüber hinaus noch erreicht werden können – z. B. dadurch, das wir *first*, *link* und *last* strikt nach der Idee des Iterationssatzes 3.2 konstruieren.

Daß diese Unterscheidung sinnvoll ist, kann z. B. dem Beispiel 4.11 aus Kapitel 4 entnommen werden.

Wir nutzen vor allem die Tatsache aus, daß im Dschungel alle Knoten genau eine Eingangskante haben.

**Satz 3.4** Sei  $G = (\Sigma, \Delta, P, S)$  eine Hypergraph-Grammatik mit Dschungel-Eigenschaft,

seien *first*, *link* und *last* Hypergraphen, und

seien  $link_{first}^o$ ,  $link_{link}^o$ ,  $link_{link}^u$  und  $link_{last}$  je  $n$  link-Knoten mit  $first \otimes link^j \otimes last \in L(G)$  für alle  $j \in \mathbb{N}_0$ .

Dann gilt:

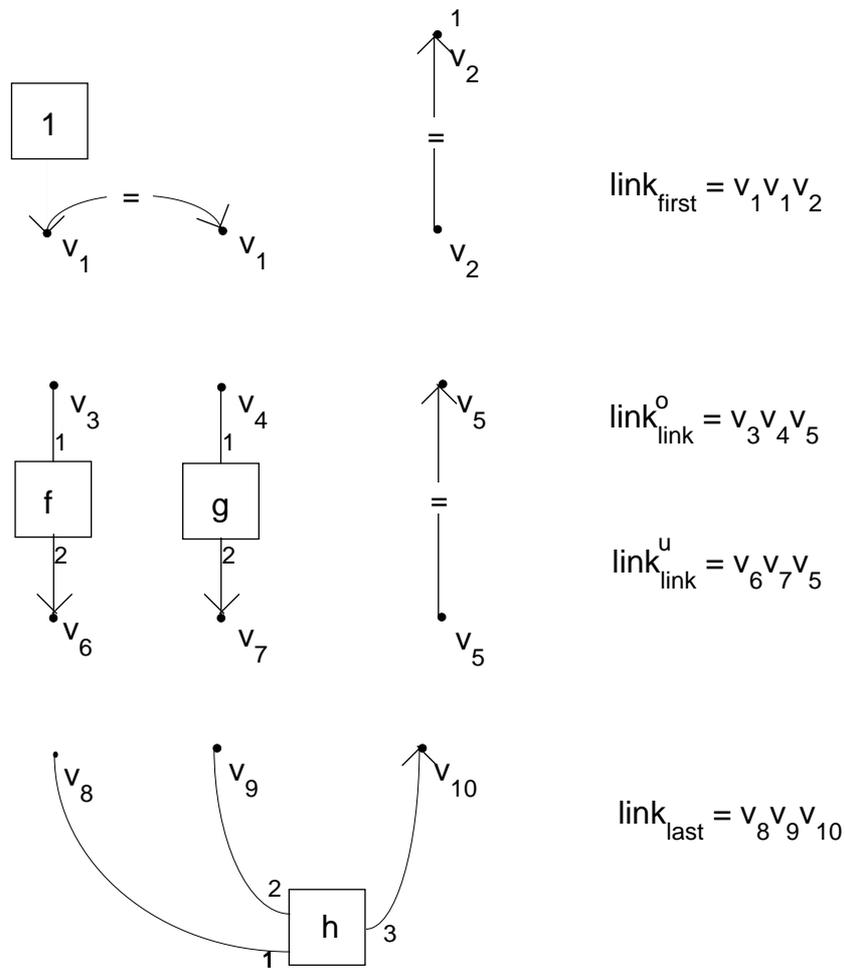


Abbildung 3.11: Andere Darstellung der Zerlegung aus Abbildung 3.9

1. Ist  $v$  ein Knoten aus  $first$  (bzw.  $link$  oder  $last$ ), der nicht link-Knoten ist, dann besitzt  $v$  in  $first$  (bzw.  $link$  oder  $last$ ) genau eine Eingangskante.
2. Für ein beliebiges, aber festes  $i$  ( $1 \leq i \leq n$ ) **entweder**
  - (a) Der  $i$ -te link-Knoten aus  $link_{link}^o$  und aus  $link_{last}$  besitzt keine Eingangskante, der  $i$ -te link-Knoten aus  $link_{first}$  besitzt genau eine Eingangskante und der  $i$ -te link-Knoten aus  $link_{link}^u$  besitzt entweder genau eine Eingangskante oder ist identisch mit dem  $j$ -ten link-Knoten aus  $link_{link}^o$  und für dieses  $j$  gilt das gleiche wie für  $i$  (in diesem Fall also 2a)).
  - Oder**
  - (b) (dual zu (a)).

**Beweis:**

1. Angenommen,  $v$  hat nicht genau eine Eingangskante. Dann hat  $v$  auch in  $H = first \otimes (link \otimes last)$  nicht genau eine Eingangskante, also ist  $H$  kein Dschungel – Widerspruch zu  $H \in L(G)$ .
2. Vorab: Hat irgendein link-Knoten mehr als eine Eingangskante, dann folgt, daß  $first \otimes (link \otimes last)$  kein Dschungel ist und damit ein Widerspruch.
  - (a) Wir nehmen an, daß  $x_i$  genau eine Eingangskante in  $first$  besitzt, aber eine der weiteren Einschränkungen nicht erfüllt ist:
    - i.  $y_i^o$  ( $i$ -ter link-Knoten aus  $link_{link}^o$ ) besitzt mindestens eine Eingangskante in  $link$ , dann besitzt  $x_i$  in  $H = first \otimes (link \otimes last)$  mindestens zwei Eingangskanten – Widerspruch zu  $H \in L(G)$ .
    - ii.  $z_i$  ( $i$ -ter link-Knoten aus  $link_{last}$ ) besitzt mindestens eine Eingangskante in  $last$ , dann besitzt  $x_i$  in  $H = first \otimes last$  mindestens zwei Eingangskanten – Widerspruch zu  $H \in L(G)$ .
    - iii. A.  $y_i^u$  ( $i$ -ter link-Knoten aus  $link_{link}^u$ ) besitzt keine Eingangskante in  $link$  und ist nicht identisch mit irgendeinem Knoten aus  $link_{link}^o$ . Dann besitzt  $y_i^u$  auch in  $H = first \otimes (link \otimes last)$  keine Eingangskante, also ist  $H$  kein Dschungel – Widerspruch zu  $H \in L(G)$ .  
B.  $y_i^u$  besitzt keine Eingangskante in  $link$ , ist aber identisch mit  $y_j^o$ , dem  $j$ -ten link-Knoten aus  $link_{link}^o$ . Daraus folgt, daß auch  $y_j^o$  keine Eingangskante in  $link$  besitzt. Wenn  $j$  sich aber „anders“ verhält als  $i$ , kippt insbesondere unsere Anfangsannahme für diesen Fall – also  $x_j$  hat keine Eingangskante in  $first$ , weswegen entweder  $x_j$  in  $H = first \otimes (link \otimes last)$  (dann gelinkt mit  $y_j^o$  alias  $y_i^u$ ) keine Eingangskante besitzt, oder

aber  $z_i$  „Ursache“ dieser Eingangskante ist – was ebenso zum Widerspruch führt.

(b) Argumentation ist dual zu der obigen.

□

Zur Beschreibung weiterer Eigenschaften benötigen wird zunächst die folgende Definition

**Definition 3.5 (auf- und abwärts-link-Knoten)** Sei  $G = (\Sigma, \Delta, P, S)$  eine Hypergraph-Grammatik mit Dschungel-Eigenschaft, seien  $first$ ,  $link$  und  $last$  Hypergraphen, und seien  $link_{first}$ ,  $link_{link}^o$ ,  $link_{link}^u$  und  $link_{last}$  je  $n$  link-Knoten mit  $first \otimes link^j \otimes last \in L(G)$  für alle  $j \in \mathbb{N}_0$ . Der  $i$ -te link-Knoten heißt abwärts-link-Knoten, wenn der  $i$ -te link-Knoten aus  $link_{first}$  eine Eingangskante in  $first$  besitzt. Er heißt aufwärts-link-Knoten, wenn der  $i$ -te link-Knoten aus  $link_{last}$  eine Eingangskante in  $last$  besitzt.

**Bemerkungen:**

1. Bei der hier gewählten Konvention,  $first$ ,  $link$  und  $last$  zu zeichnen, befindet sich die Eingangskante zu einem abwärts-link-Knoten oberhalb des link-Knotens – die Information fließt dort sozusagen abwärts. Umgekehrtes gilt für aufwärts-link-Knoten.

2. Bei der „unnatürlichen“ Zerlegung aus Abbildung 4.9 (Kapitel 4) sind der dritte und der vierte link-Knoten weder auf- noch abwärts-link-Knoten. Es wäre durchaus möglich, die Definition so zu erweitern, daß – wie es die Abbildung 4.9 suggeriert – der dritte link-Knoten ein abwärts-link-Knoten und der vierte ein aufwärts-link-Knoten ist.

Da dies aber der Übersichtlichkeit abträglich ist und bei einer „natürlichen“ Zerlegung solche Knoten nicht auftreten (wie wir gleich noch zeigen werden), ist davon Abstand genommen worden.

Betrachten wir nun noch einmal die Hypergraphen aus Abbildung 3.5:

Sei  $e_1$  die explizit gezeichnete Hyperkante mit dem Label  $A$  aus  $H_1$ .

Da  $H_1$  ein Dschungel ist, muß für die Knoten  $nod(e_1, 1), \dots, nod(e_1, n - 1)$  gelten, daß sie in  $H_1 - A$  (dem späteren  $first$ ) eine Eingangskante besitzen; der Knoten  $nod(e_1, n)$  hingegen besitzt  $e_1$  als Eingangskante und darf daher keine Eingangskante in  $H_1 - A$  haben.

Für  $first$  folgt also, daß genau die ersten  $n - 1$  link-Knoten eine Eingangskante in  $first$  besitzen.

Sei  $e_2$  die explizit gezeichnete Hyperkante mit dem Label  $A$  aus  $H_2$ .

Da  $(H_1 - A) \otimes H_2$  ein Dschungel ist, darf der Knoten  $nod(e_2, n)$  außer  $e_2$  keine weiteren Eingangskanten besitzen. Ersetzt man  $e_2$  durch  $H_3$ , erhält man wiederum einen Dschungel, also muß die Eingangskante des genannten Knotens sich in  $H_3$  – dem späteren  $last$  – befinden.

Damit ist (nach Anwendung von Satz 3.4) das Verhalten aller link-Knoten (bei Konstruktion gemäß der Idee des Iterationssatzes) geklärt und wir können folgern:

**Folgerung 3.6** *Sei  $G = (\Sigma, \Delta, P, S)$  eine Hypergraph-Grammatik mit Dschungel-Eigenschaft und*

*$H$  ein hinreichend großer Hypergraph aus  $L(G)$ .*

*Dann existieren Hypergraphen  $first$ ,  $link$  und  $last$  mit*

*link-Knoten  $link_{first}$ ,  $link_{link}^o$ ,  $link_{link}^u$  und  $link_{last}$  so, daß*

*$first \otimes link \otimes last = H$ ,*

*$first \otimes link^i \otimes last \in L(G)$  für alle  $i \in \mathbb{N}_0$ , und (sei  $n$  die Anzahl der link-Knoten)*

*1. die ersten  $n - 1$  link-Knoten sind abwärts-link-Knoten.*

*2. der  $n$ -te link-Knoten ist ein aufwärts-link-Knoten.*

**Bemerkung:**

Als zusätzliche Eigenschaft folgt, daß dann der  $n$ -te link-Knoten von  $link_{first}$  bzw.  $link_{link}^u$  disjunkt von allen anderen ist.

# Kapitel 4

## Lineare Hypergraph-Grammatiken

### 4.1 Formen der Linearität

Betrachtet man kontextfreie Grammatiken, dann werden dort drei Formen der „Linearität“ unterschieden: linear, rechtslinear und linkslinear. Rechtslineare und linkslineare Grammatiken erzeugen die gleiche Sprachfamilie, die wiederum echte Teilmenge der Sprachfamilie der durch lineare Grammatiken erzeugten Sprachen ist.

Für Hypergraph-Grammatiken gibt es in dem Sinne kein links und rechts. Da wir aber unsere Dschungel meist so dargestellt haben, daß der (höchste) externe Knoten sich ganz oben befindet, können wir für kontextfreie Hypergraph-Grammatiken mit Dschungel-Eigenschaft eine ähnliche Dreiteilung vornehmen: linear, obenlinear und untenlinear.

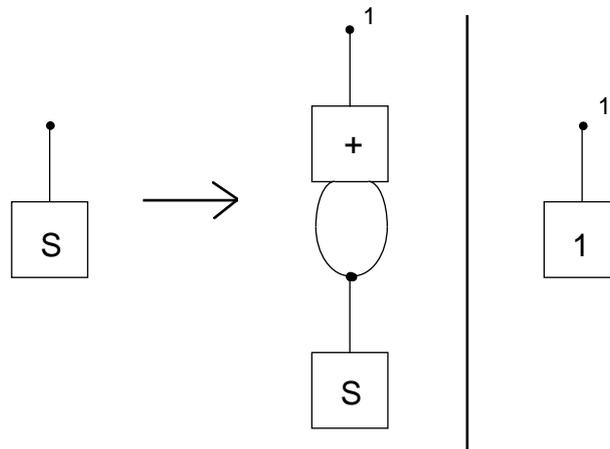
Wir werden zeigen, daß in diesem Fall die beiden speziellen Formen der Linearität – also obenlinear und untenlinear – nicht die gleiche Sprachfamilie erzeugen.

J. Engelfriet und G. Leih haben in [EnLe89] bereits nachgewiesen, daß lineare Hypergraph-Grammatiken nicht alle durch kontextfreie Hypergraph-Grammatiken erzeugbaren Sprachen erzeugen können. Ihre Argumentation gilt auch bei Berücksichtigung der Dschungel-Eigenschaft, daher wird hier darauf nicht weiter eingegangen.

**Definition 4.1 (linear, obenlinear, untenlinear)** *Eine kontextfreie Hypergraph-Grammatik  $G = (\Sigma, \Delta, P, S)$  mit Dschungel-Eigenschaft heißt*

**linear**, falls für jede Produktion  $p = (B, H_B) \in P$  gilt:  $|H_B|_N \leq 1$  (die Zahl der Hyperkanten mit nichtterminalem Label ist kleiner als 2).

**obenlinear**, falls sie linear ist und außerdem für jede Produktion  $p = (B, H_B) \in P$  gilt: Falls  $|H_B|_N = 1$  ist, dann hat die Eingangskante des höchsten externen Knotens das nichtterminale Label.

Abbildung 4.1: Produktionen aus  $P_u$ 

**untenlinear**, falls sie linear ist und außerdem für alle  $B \in \Sigma - \Delta$  gilt:  $\text{rank}_\Sigma(B) = 1$ .

**Bemerkung:**

Einzig die Definition von obenlinear muß geändert werden, will man diese Definition auf kontextfreie Hypergraph-Grammatiken ohne Dschungel-Eigenschaft ausdehnen – dort stören 0-Hyperkanten und mehrere Eingangskanten.

Eine kontextfreie Hypergraph-Grammatik heißt also linear, wenn auf der rechten Seite einer Produktion nicht mehr als ein nichtterminales Label auftauchen darf. Sie heißt obenlinear, wenn dieses Label, so vorhanden, sich „ganz oben“ befindet, wobei oben, wie bereits eingangs erwähnt, im Dschungel durch den externen Knoten festgelegt wird.

Die Forderung  $\text{rank}_\Sigma(B) = 1$  für untenlineare Grammatiken führt dazu, daß die mit  $B$  markierte Hyperkante keine Vorgänger besitzt, also keine andere Hyperkante direkt unter dieser einzuzeichnen wäre.

Intuitiv drängt sich der Verdacht auf, daß eine Restriktion des Ranges wie im Fall der untenlinearen Hypergraph-Grammatik die massivste der hier vorgenommenen Beschränkungen ist. In der Tat werden wir später zeigen, daß dies so ist.

Doch zunächst geben wir zur weiteren Illustration zu jedem dieser drei Typen ein kleines Beispiel.

**Beispiel 4.2** Die Grammatik  $G_u = (\{+, 1, S\}, \{+, 1\}, P_u, S)$ ,  $\text{rank}(S) = \text{rank}(1) = 1$ ,  $\text{rank}(+) = 3$ , mit den Produktionen aus Abbildung 4.1 ist untenlinear, aber nicht obenlinear.

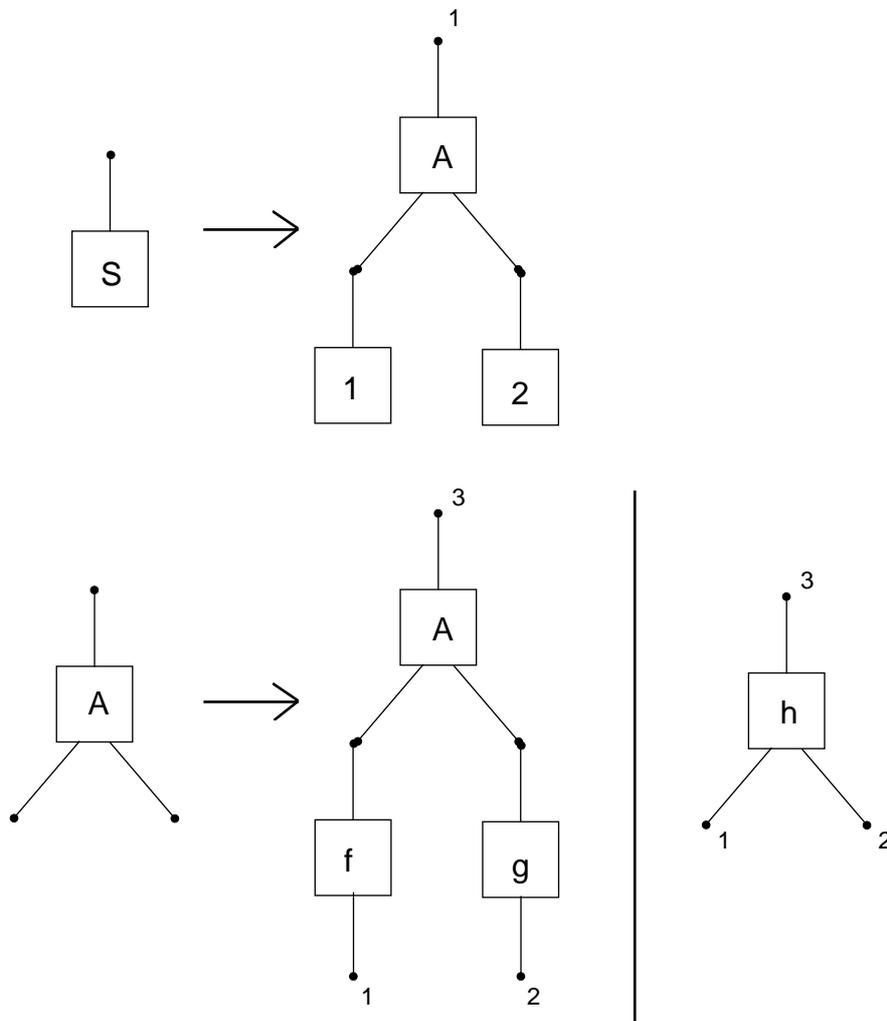
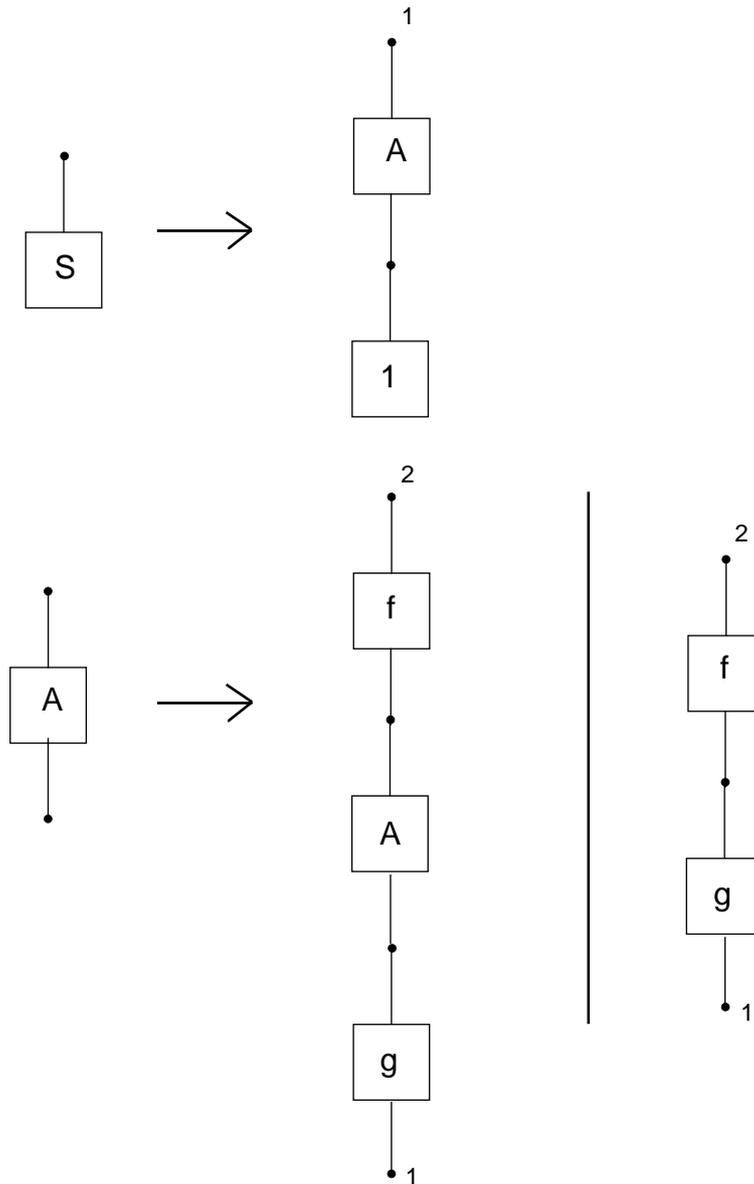


Abbildung 4.2: Produktionen aus  $P_o$

**Beispiel 4.3** Die Grammatik  $G_o = (\{1, 2, f, g, h, A, S\}, \{1, 2, f, g, h\}, P_o, S)$ ,  $rank(S) = rank(1) = rank(2) = 1$ ,  $rank(f) = rank(g) = 2$ ,  $rank(A) = rank(h) = 3$ , mit den Produktionen aus Abbildung 4.2 ist obilinear, aber nicht untenlinear.

**Beispiel 4.4** Die Grammatik  $G_l = (\{1, f, g, A, S\}, \{1, f, g\}, P_l, S)$ ,  $rank(S) = rank(1) = 1$ ,  $rank(A) = rank(f) = rank(g) = 2$ , mit den Produktionen aus Abbildung 4.3 ist linear, aber weder obilinear noch untenlinear.

Betrachten wir Ableitungsbäume zu linearen Hypergraph-Grammatiken, so ist klar, daß diese sehr einfach aufgebaut sind – es handelt sich sozusagen um einen Strang oder eine Kette. Außerdem ist die Anzahl der Hyperkanten mit nichtterminalen Label in einer nichtterminalen Satzform einer solchen Grammatik immer eins.

Abbildung 4.3: Produktionen aus  $P_l$

Das heißt insbesondere, daß, wenn die lineare Hypergraph-Grammatik  $G = (\Sigma, \Delta, P, S)$   $n$  verschiedene nichtterminale Symbole besitzt ( $n = |\Sigma - \Delta|$ ), es spätestens nach  $n + 1$  Ableitungsschritten ein Symbol  $A$  gibt, das wiederholt abgeleitet wurde.

Ist  $m$  die maximale Anzahl von Hyperkanten auf der rechten Seite einer Produktion aus  $G$ , läßt sich die Zahl der zwischen dem ersten und dem zweiten Ableiten dieses Symbols  $A$  zusätzlich erzeugten Hyperkanten nach oben abschätzen:  $n * (m - 1)$ . Denn wir ersetzen in einem Ableitungsschritt eine Hyperkante durch höchstens  $m$  Hyperkanten – gewinnen also  $m - 1$  Hyperkanten hinzu – und spätestens mit dem  $n$ -ten Ableitungsschritt produzieren wir ein nichtterminales Symbol, das zuvor bereits einmal abgeleitet wurde. Diese Betrachtung können wir ab der Wurzel machen – also ganz oben im Ableitungsbaum, denn in der Informatik wachsen die Bäume ja nicht in den Himmel – und kommen somit zu der

**Folgerung 4.5** Sei  $G = (\Sigma, \Delta, P, S)$  eine lineare Hypergraph-Grammatik,  $n = |\Sigma - \Delta|$ ,  $m = \max\{|H'| \mid p = (B, H') \in P\}$  und ist  $H \in L(G)$  mit  $|H| > n * (m - 1) + 1$ . Dann gibt es eine Zerlegung von  $H$  in *first*, *link* und *last* gemäß Iterationssatz, so daß  $|first| + |link| \leq n * (m - 1)$  ist.

Betrachten wir hierzu das folgende Beispiel:

**Beispiel 4.6** Es sei  $G_{il} = (\{1, f, A, B, S\}, \{1, f\}, P_{il}, S)$ ,  $\text{rank}(S) = \text{rank}(1) = \text{rank}(A) = \text{rank}(B) = 1$ ,  $\text{rank}(f) = 2$ , mit den Produktionen aus Abbildung 4.4.

Es ist  $n = 3$ ,  $m = 5$  und gemäß unserer Überlegungen erhalten wir für „ $f^{36}(1)$ “: (siehe Abbildung 4.5)

Unser *link* enthält  $12 = 3 * (5 - 1)$  Knoten. Da die durch  $G_{il}$  festgelegten Terme alle von der Form  $f^{12*n}(1)$ ,  $n \geq 0$ , sind, haben wir gleichzeitig den kleinstmöglichen (nicht trivialen) *link* gewählt.

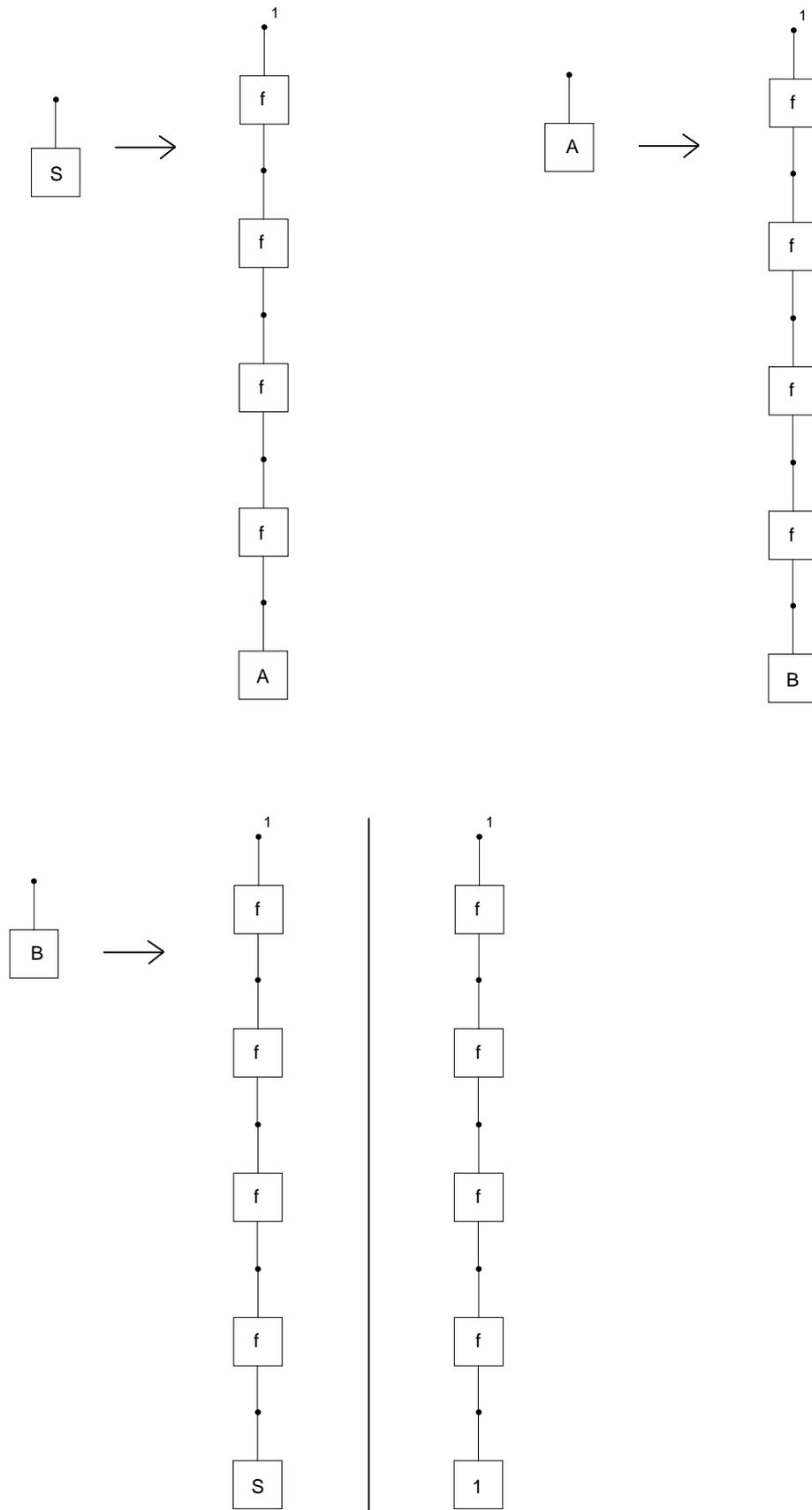
An dieser Stelle wollen wir kurz eine „ganz ähnliche“, aber nicht lineare kontextfreie Hypergraph-Grammatik zum Vergleich betrachten:

**Beispiel 4.7** Es sei  $G_k = (\{1, f, A, B, S\}, \{1, f\}, P_k, S)$ ,  $\text{rank}(S) = \text{rank}(1) = 1$ ,  $\text{rank}(f) = \text{rank}(A) = \text{rank}(B) = 2$ , mit den Produktionen aus Abbildung 4.6.

In Abbildung 4.7 ist eine Ableitung angegeben.

Die Anzahl der nichtterminalen Symbole und die maximale Anzahl der Hyperkanten ist wie im Beispiel 4.6, die durch diese Grammatik festgelegten Terme sind von der Form  $f^{100*n}(1)$ ,  $n \geq 0$ , der kleinstmögliche *link* umfaßt also 100 Hyperkanten statt 12.

Es ist  $100 = 4 * 5^2 = (m - 1) * m^{(n-1)} < m^n$ , das Beispiel läßt sich entsprechend erweitern.

Abbildung 4.4: Produktionen aus  $P_l$

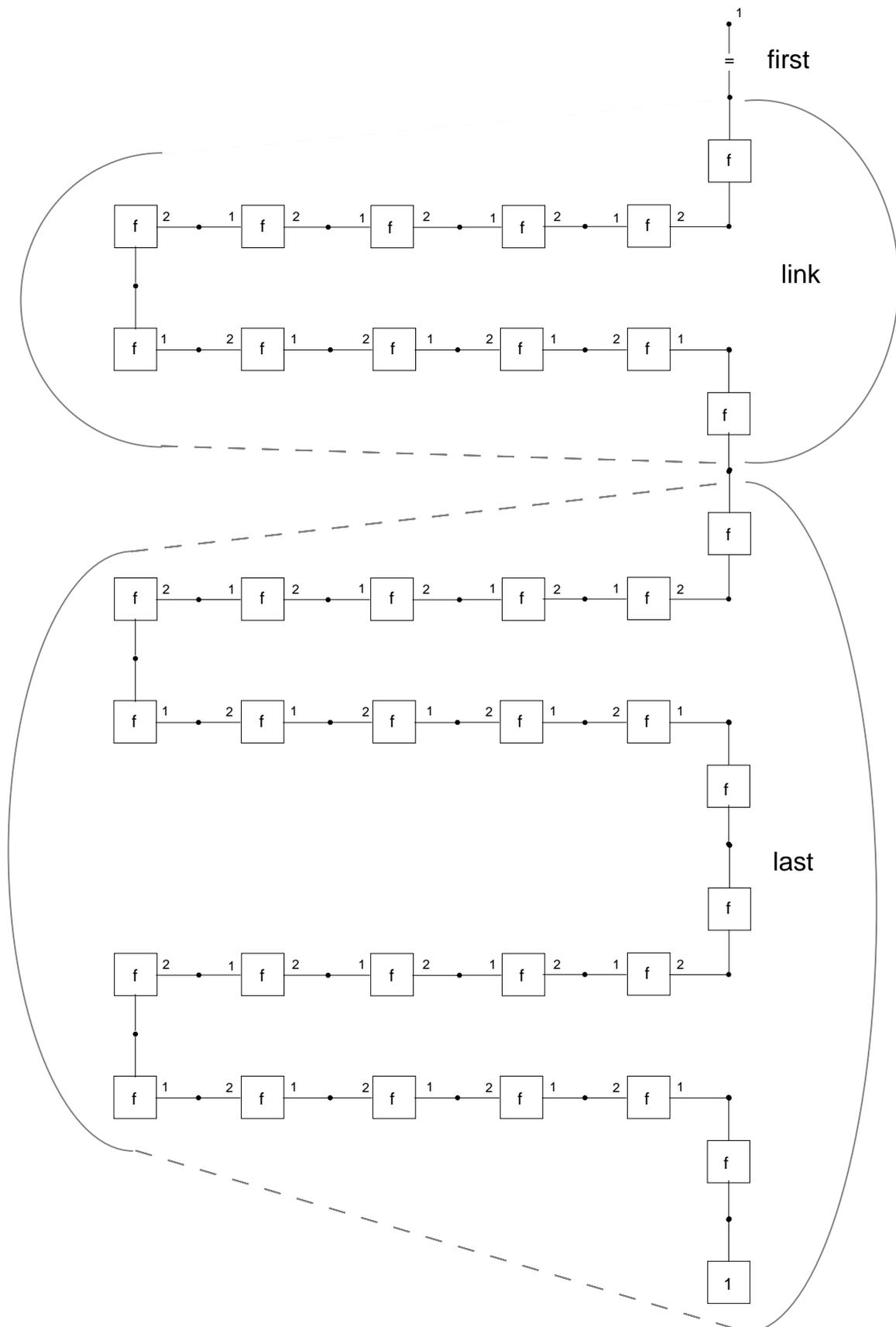
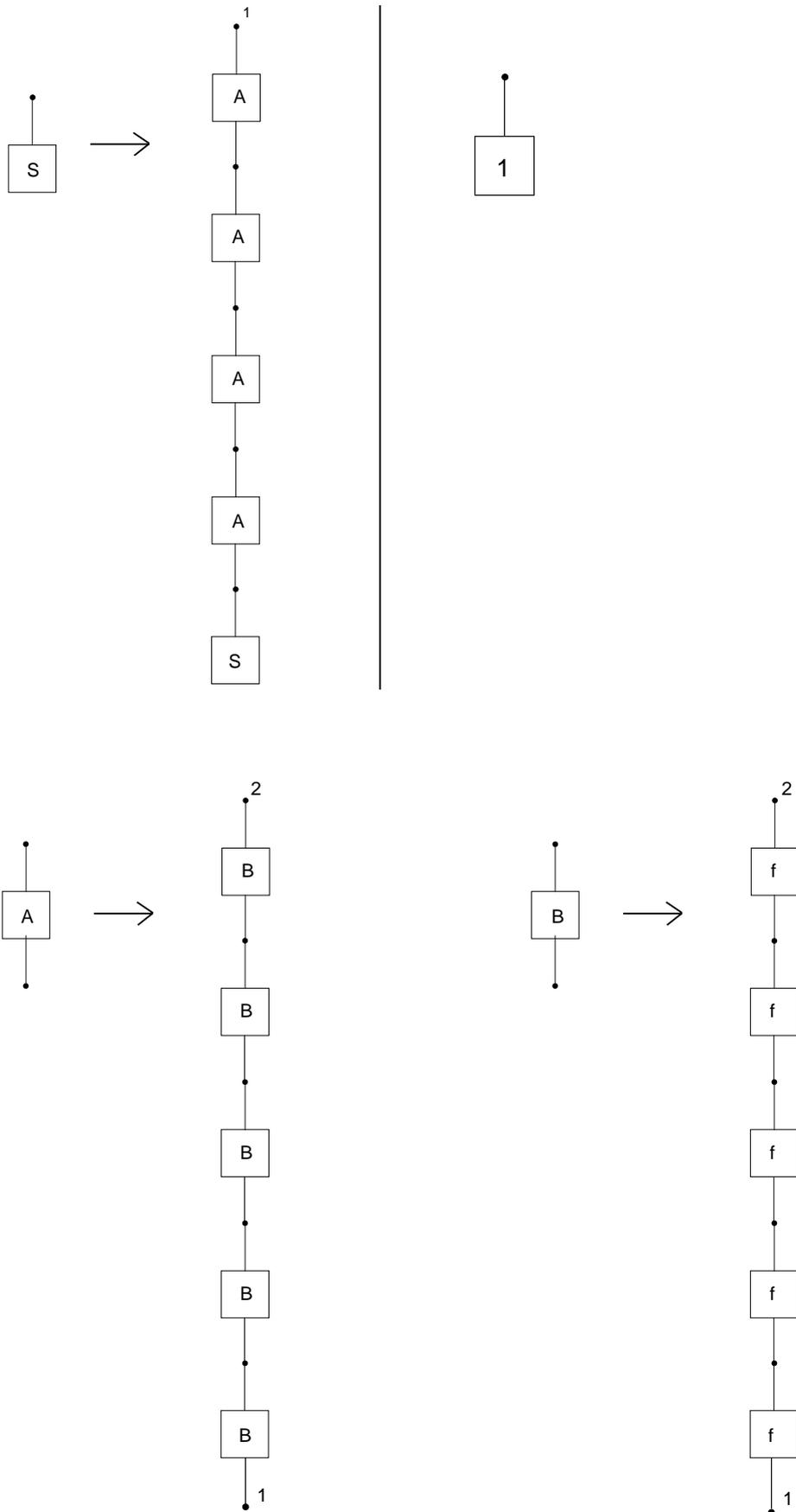


Abbildung 4.5: Zerlegung von „ $f^{36}(1)$ “

Abbildung 4.6: Produktionen aus  $P_k$

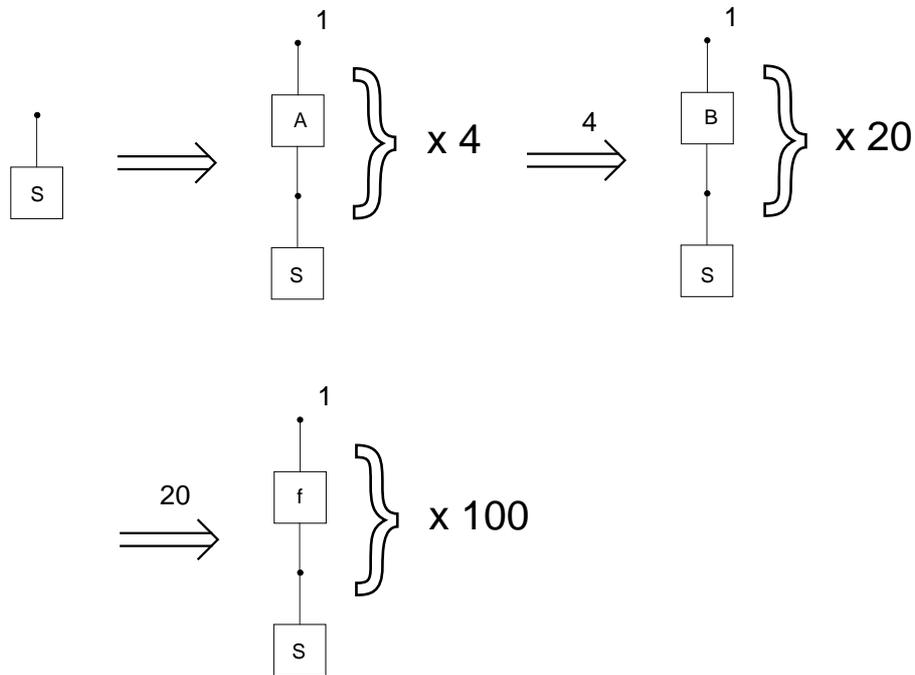


Abbildung 4.7: Ableitung in  $G_k$

Kehren wir zu den Betrachtungen zurück, die zur Folgerung 4.5 führten, und wenden wir uns dem unteren Ende des Ableitungsbaums zu, dann können wir ähnliche Schlüsse ziehen. Allerdings kann die letzte Produktion, da sie terminal ist, nicht zum *link* „beitragen“, weswegen ein Ableitungsschritt mehr kalkuliert werden muß.

**Folgerung 4.8** Sei  $G = (\Sigma, \Delta, P, S)$  eine lineare Hypergraph-Grammatik,  $n = |\Sigma - \Delta|$ ,  $m = \max\{|H'| \mid p = (B, H') \in P\}$  und ist  $H \in L(G)$  mit  $|H| > n * (m - 1) + 1$ .

Dann gibt es eine Zerlegung von  $H$  in *first*, *link* und *last* gemäß Iterationssatz, so daß

$$|link| + |last| \leq n * (m - 1) + m = (n + 1) * (m - 1) + 1 \text{ und} \\ |link| \leq n * (m - 1) \text{ ist.}$$

In beiden Folgerungen ist  $|link| \leq n * (m - 1)$ , im ersten Fall muß die Starthyperkante, im zweiten die „Schlußproduktion“ mit  $m$  Hyperkanten bedacht werden.

Nun stellt sich die Frage, ob für oben- und untenlineare Hypergraph-Grammatiken weitere Einschränkungen getroffen werden können.

Offenbar sind bezüglich der Größe der *first*, *link* und *last* weitere Einschränkungen so nicht möglich (unser Beispiel 4.6 war untenlinear und linear, für obenlineare kann man unter der zusätzlichen Bedingung, daß die Grammatik die reiner-Dschungel-

Eigenschaft besitzt, etwas weiter einschränken, da dann das Startsymbol  $S$  nicht auf der rechten Seite auftreten darf (außer  $S \rightarrow S$ , was kein Gewinn ist)).

Allerdings ist es in allen nicht terminalen Satzformen einer obenlinearen Hypergraph-Grammatik so, daß die einzige Hyperkante mit nichtterminalem Label Eingangskante des (höchsten) externen Knotens ist.

**Satz 4.9** Sei  $G = (\Sigma, \Delta, P, S)$  eine obenlineare Hypergraph-Grammatik,

$n = |\Sigma - \Delta|$ ,  $m = \max\{|H'| \mid p = (B, H') \in P\}$ ,

$l = \max\{\text{rank}_\Sigma(B) \mid B \in \Sigma - \Delta\}$ , und sei

$H \in L(G)$  mit  $|H| > n * (m - 1) + 1$ .

Dann gibt es eine Zerlegung von  $H$  in *first*, *link* und *last* gemäß Iterationssatz mit

$k$  link-Knoten ( $\text{link}_{\text{first}} = x_1^f \dots x_k^f$ ,  $\text{link}_{\text{link}}^{\text{oben}} = x_1^o \dots x_k^o$ ,  $\text{link}_{\text{link}}^{\text{unten}} = x_1^u \dots x_k^u$ ), so daß

$|\text{first}| + |\text{link}| \leq n * (m - 1)$ ,

$k \leq l$ ,

in *first* ist der externe Knoten identisch mit  $x_k^f$ ,

in *link* ist  $x_k^o$  identisch mit  $x_k^u$  und

alle anderen link-Knoten sind abwärts-link-Knoten.

**Beweis:**

$H \in L(G) \Rightarrow$  es existiert ein Ableitungsbaum (eine -kette) bezüglich  $G$ , deren Auswertung  $H$  ergibt.

Aus der Größe von  $H$  folgt, daß der Ableitungsbaum mindestens  $n + 1$  Knoten hat, also mindestens ein nichtterminales Symbol mehr als einmal abgeleitet wurde.

Wir betrachten die obersten  $n$  Knoten des Ableitungsbaums und konstruieren daraus *first* und *link* gemäß Iterationssatz.

Dann ist  $|\text{first}| + |\text{link}| \leq n * (m - 1)$  und, da  $G$  obenlinear ist, gelten die genannten Identitäten. □

Wozu die Forderung  $k \leq l$  dient, wird an dem folgenden Beispiel verdeutlicht.

**Beispiel 4.10** Für die lineare (aber nicht obenlineare) Hypergraph-Grammatik  $G_l$  aus Beispiel 4.4 (Abbildung 4.3) sieht die Zerlegung von  $f^2(g^2(1))$  gemäß Iterationssatz wie in Abbildung 4.8 aus.

Von Hand kann man diese z. B. wie in Abbildung 4.9 modifizieren. Diese Zerlegung ist zwar nicht „ablesbar“ aus der Grammatik, aber ganz offensichtlich erfüllt sie die Bedingungen des Iterationssatzes.

Im Prinzip sind die beiden rechten link-Knoten redundant, lassen sich aber gerade deswegen entsprechend in jede *first*, *link*, *last* Zerlegung einbauen und würden daher den Satz 4.9 entwerten – wenn da nicht die Beschränkung ( $k \leq l$ ) wäre.

Umgekehrt kann es für Sprachen, die durch obenlineare Hypergraph-Grammatiken erzeugt werden, durchaus Zerlegungen mit „kleinen“ *first* und *link* geben, die nicht

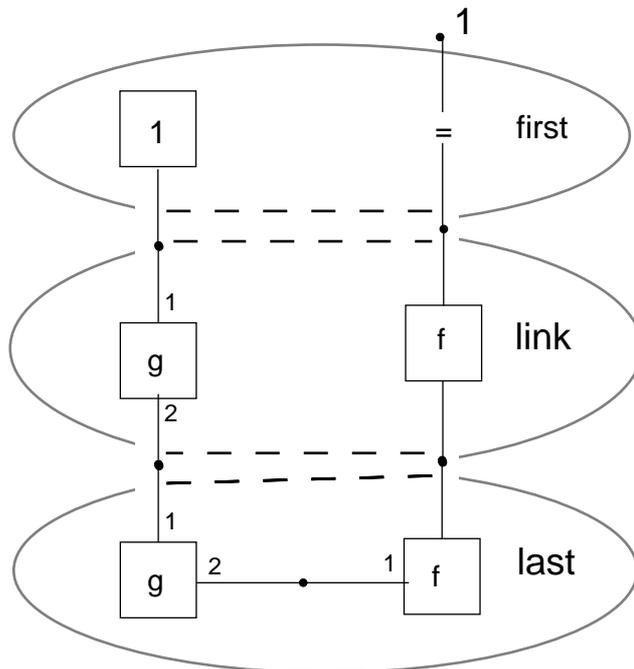


Abbildung 4.8: Zerlegung von „ $f^2(g^2(1))$ “  $\in L(G_l)$

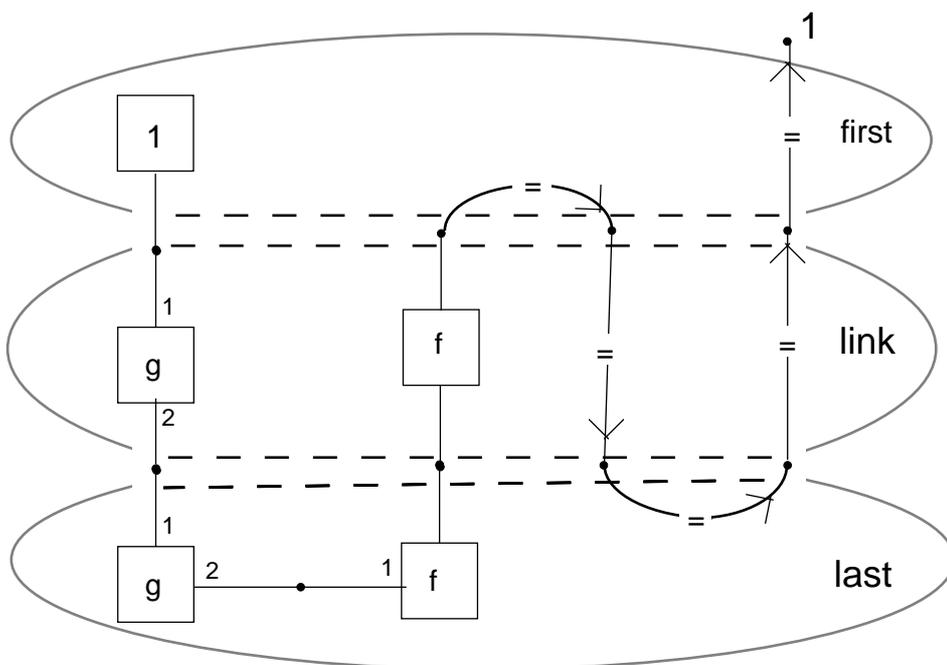


Abbildung 4.9: „Andere“ Zerlegung von „ $f^2(g^2(1))$ “  $\in L(G_l)$

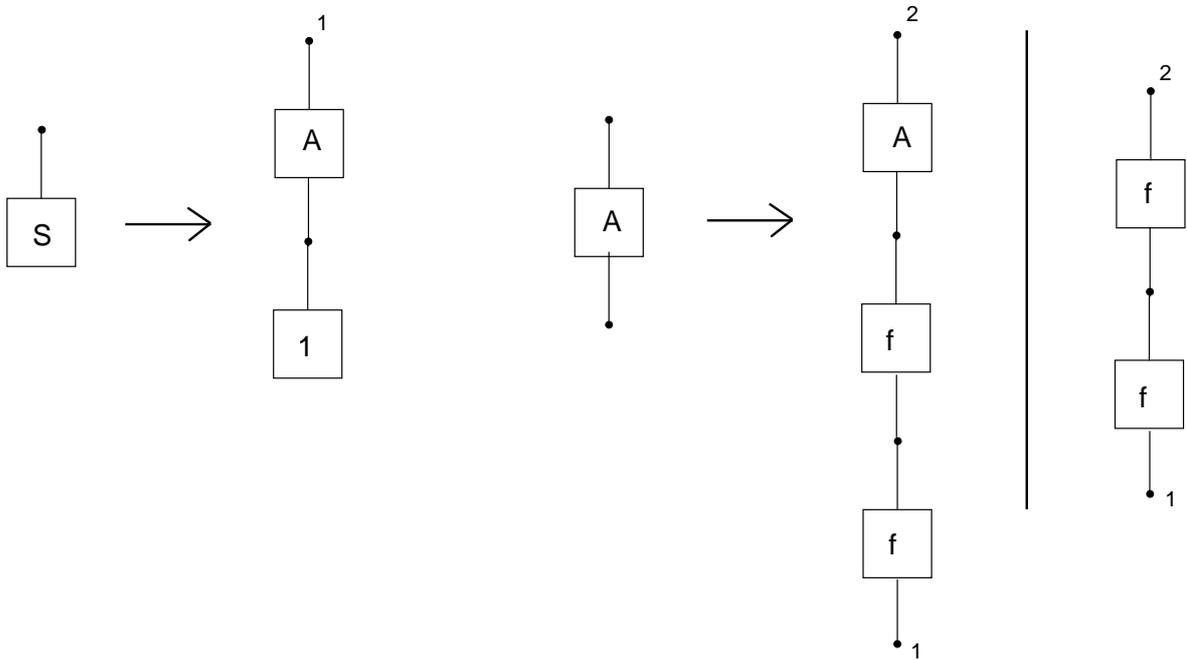


Abbildung 4.10: Produktionen aus  $P$

die Eigenschaft haben, daß der rechteste link-Knoten identifiziert wird. Auch hierzu ein kleines Beispiel:

**Beispiel 4.11** Sei  $G = (\{f, 1, A, S\}, \{f, 1\}, P, S)$  mit  $P$  wie in Abbildung 4.10. Dann ist in Abbildung 4.11 die Zerlegung gemäß Iterationssatz für  $f^4(1)$  angegeben.

Aber auch die Zerlegungen aus den Abbildungen 4.12, 4.13 und 4.14 (und weitere) sind möglich.

Für untenlineare Hypergraph-Grammatiken gestaltet sich die Iterationsmöglichkeit insofern recht einfach, als daß man mit nur einem link-Knoten auskommt.

**Satz 4.12** Sei  $G = (\Sigma, \Delta, P, S)$  eine untenlineare Hypergraph-Grammatik,  $n = |\Sigma - \Delta|$ ,  $m = \max\{|H'| \mid p = (B, H') \in P\}$  und ist  $H \in L(G)$  mit  $|H| > n * (m - 1) + 1$ .

Dann gibt es eine Zerlegung von  $H$  in first, link und last gemäß Iterationssatz mit nur einem link-Knoten. Dieser ist ein aufwärts-link-Knoten.

**Beweis:**

$H \in L(G) \Rightarrow$  es existiert ein Ableitungsbaum (eine -kette) bezüglich  $G$ , deren

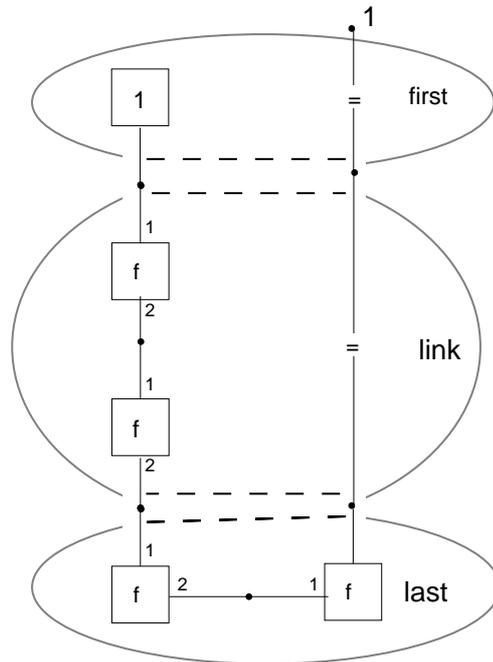


Abbildung 4.11: Zerlegung von „ $f^4(1)$ “  $\in L(G)$

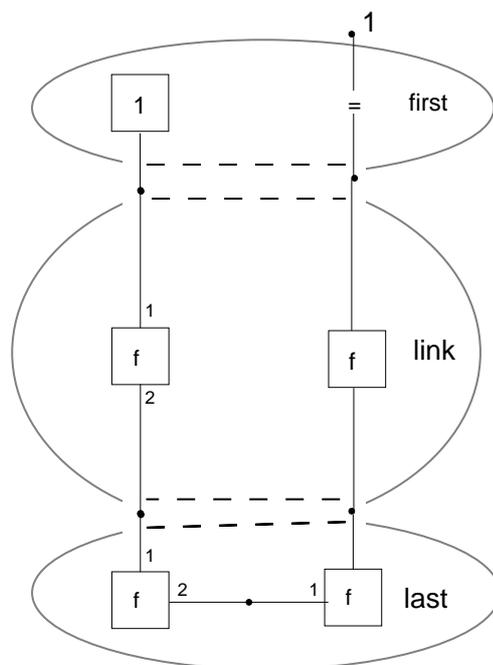
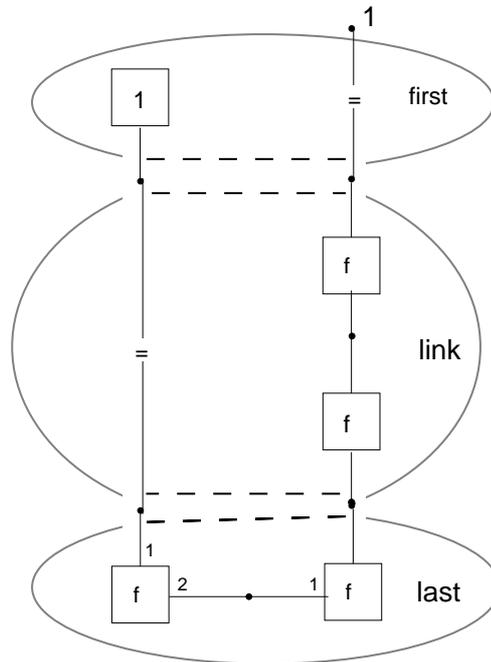
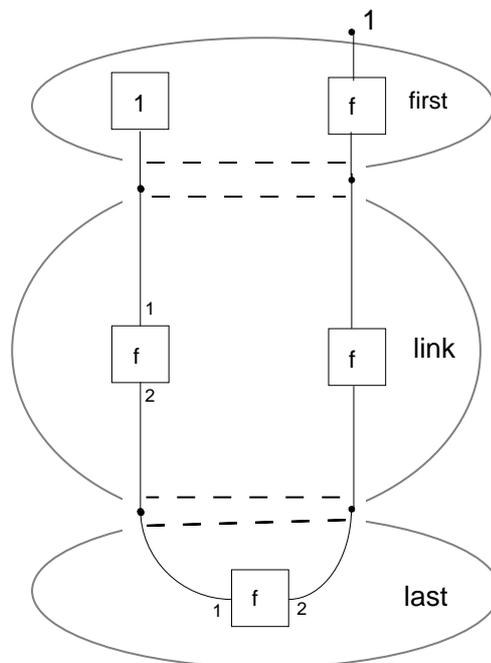


Abbildung 4.12: „Andere“ Zerlegung von „ $f^4(1)$ “

Abbildung 4.13: „Andere“ Zerlegung von „ $f^4(1)$ “Abbildung 4.14: „Andere“ Zerlegung von „ $f^4(1)$ “

Auswertung  $H$  ergibt.

Aus der Größe von  $H$  folgt, daß der Ableitungsbaum mindestens  $n + 1$  Knoten hat, also mindestens ein nichtterminales Symbol  $A$  wiederholt abgeleitet wurde.

Wir betrachten die obersten  $n$  Knoten des Ableitungsbaums und konstruieren daraus *first* und *link* gemäß Iterationssatz.

Dann ist  $|first| + |link| \leq n * (m - 1)$ , und wegen  $rank(A) = 1$  gibt es genau einen link-Knoten.

□

## 4.2 Sprachfamilien

Wir wollen untersuchen, wie die durch lineare, obenlineare und untenlineare Hypergraph-Grammatiken festgelegten Sprachfamilien zueinander stehen.

### Definition 4.13

$\mathcal{L}_{linear} := \{L \mid \text{es existiert eine lineare Hypergraph-Grammatik } G \text{ mit } L(G) = L\}$

$\mathcal{L}_{obenlineare} := \{L \mid \text{es existiert eine obenlineare}$

$\text{Hypergraph-Grammatik } G \text{ mit } L(G) = L\}$

$\mathcal{L}_{untenlineare} := \{L \mid \text{es existiert eine untenlineare}$

$\text{Hypergraph-Grammatik } G \text{ mit } L(G) = L\}$

Obenlineare Hypergraph-Grammatiken sind eine spezielle Form der linearen Hypergraph-Grammatiken. Insofern ist klar, daß die Sprachfamilie, die durch erstere erzeugt wird, Teilmenge jener Sprachfamilie ist, die durch letztere erzeugt wird. Es stellt sich nun die Frage, ob es sich um eine echte Teilmenge handelt.

Kehren wir zu dem Beispiel 4.4 zurück (das ja ein wenig an die lineare Sprache  $a^n b^n$  erinnert) und nehmen wir einmal an, wir hätten eine obenlineare Hypergraph-Grammatik  $G = (\Sigma, \Delta, P, S)$ , die die gleiche Sprache erzeugt.

Dann könnten wir einen hinreichend großen Hypergraphen  $H \in L(G)$  gemäß Satz 4.9 so zerlegen, daß wir möglichst kleine Hypergraphen *first* und *link* und wenige link-Knoten haben (und möglicherweise einen großen Hypergraphen *last*).

Das Problem ist das „Einhängen“ weiterer  $f$ 's (mittels *link*), da hierfür nach Satz 4.9 ausschließlich abwärts-link-Knoten zur Verfügung stehen:

Von jeder Hyperkante muß ein Pfad zum externen Knoten führen, also müssen die Hyperkanten und Knoten in *last* eine Kette bilden (es steht eben kein aufwärts-link-Knoten zur Verfügung, um eine „Schleife“ über *link* und *first* zu laufen). Da aber auch das Ergebnis der Iteration wieder eine Kette ergeben muß, kann man nicht einen Knoten aus dem „Inneren“ der Kette in *last* als link-Knoten verwenden und dort über *link* weitere Hyperkanten und Knoten anhängen (sonst hätte man eine Verzweigung).

Bleiben also nur die beiden Enden dieser Kette in *last*. „Hinten“ kann man gut  $g$ 's anhängen, „vorne“ wird über den einzigen aufwärts-link-Knoten die Verbindung zum externen Knoten hergestellt. Dort aber können nach Satz 4.9 keine  $f$ 's mehr eingefügt werden.

Also ist unsere Annahme widerlegt.

**Folgerung 4.14**  $L(G_l) \notin \mathcal{L}_{obenlinear}$ .

Wenden wir uns der Sprache  $L(G_o)$  (siehe Beispiel 4.3) zu und nehmen wir einmal an, es gäbe eine untenlineare Hypergraph-Grammatik  $G$ , die diese Sprache erzeugt. Nach Satz 4.12 gibt es dann eine Zerlegung eines hinreichend großen Hypergraphen  $H \in L(G)$  in *first*, *link* und *last* mit nur einem einzigen link-Knoten. Dann kann aber beim „Iterieren“ nur eine Kette wachsen – womit offensichtlich ein Hypergraph entsteht, der nicht zur Sprache gehört, also ein Widerspruch zur Annahme.

**Folgerung 4.15**  $L(G_o) \notin \mathcal{L}_{untenlinear}$ .

Umgekehrt ist es möglich, zu einer gegebenen untenlinearen Hypergraph-Grammatik  $G'$  eine obenlineare Hypergraph-Grammatik  $G$  mit  $L(G) = L(G')$  zu konstruieren (das Verfahren ist ganz ähnlich der Umwandlung von rechts- in linkslineare Grammatiken):

Sei  $G'' = (\Sigma'', \Delta'', P'', S)$  eine untenlineare Hypergraph-Grammatik mit  $L(G'') = L(G')$  und den beiden zusätzlichen Eigenschaften, daß erstens  $S$  nicht auf der rechten Seite einer Produktion auftritt und zweitens keine singulären Produktionen vorhanden sind (Begrifflichkeit und Beseitigung unliebsamer Produktionen wie bei linearen Grammatiken) – störend sind eigentlich nur die singulären Produktionen mit  $S$  auf der linken Seite.

Dann ist  $G = (\Sigma, \Delta, P, S)$  mit

- $rank_{\Sigma}(S) = 1$
- $rank_{\Sigma}(A) = 2$  für  $A \in (\Sigma - \Delta) \setminus \{S\}$
- $rank_{\Sigma}(a) = rank_{\Sigma''}(a)$  für  $a \in \Delta$  und
- zu jeder Produktion  $p''$  aus  $P''$  wird eine Produktion  $p$  nach dem in den Abbildungen 4.15, 4.16, 4.17 und 4.18 skizzierten Verfahren konstruiert.

Womit wir zu dem Schluß kommen

**Satz 4.16**  $\mathcal{L}_{untenlinear} \subset \mathcal{L}_{obenlinear} \subset \mathcal{L}_{linear}$ .

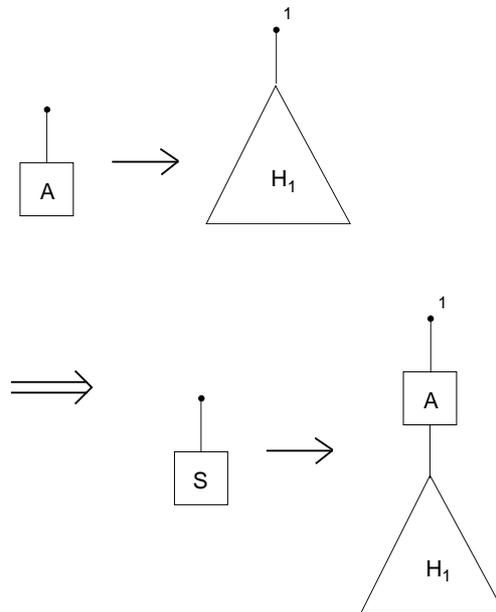


Abbildung 4.15: Ersetzungsregel 1 zur Umwandlung von untenlinear nach obenlinear

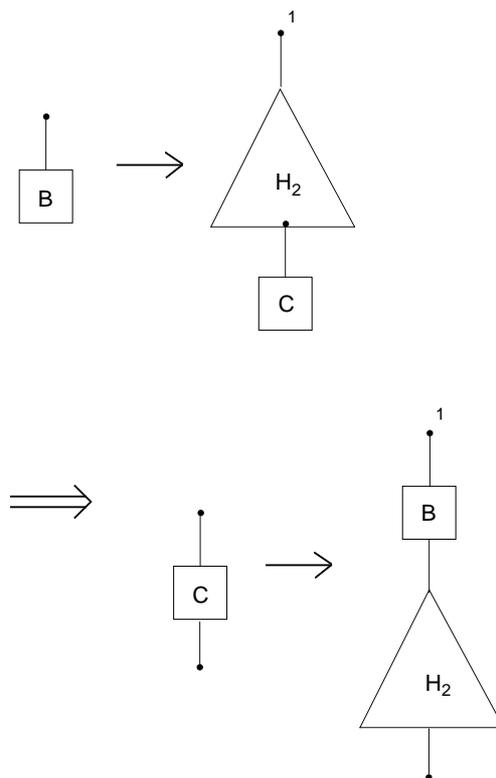


Abbildung 4.16: Ersetzungsregel 2 zur Umwandlung von untenlinear nach obenlinear

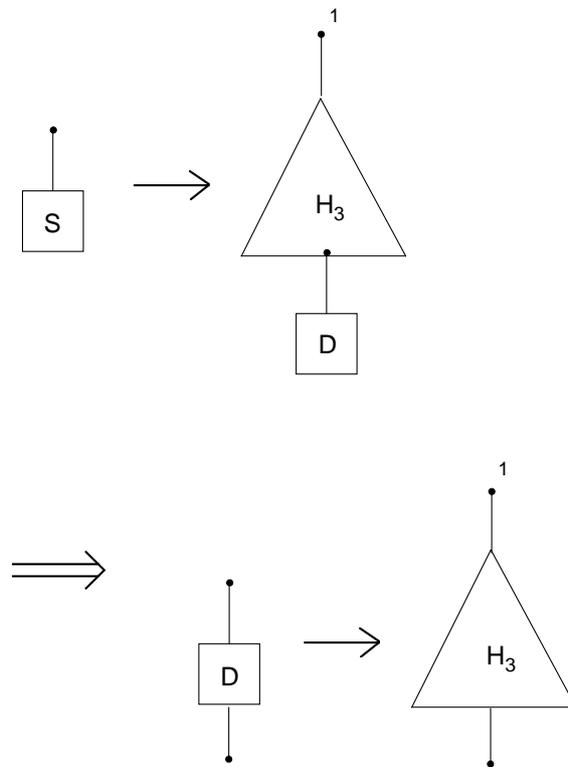


Abbildung 4.17: Ersetzungsregel 3 zur Umwandlung von untenlinear nach obenlinear

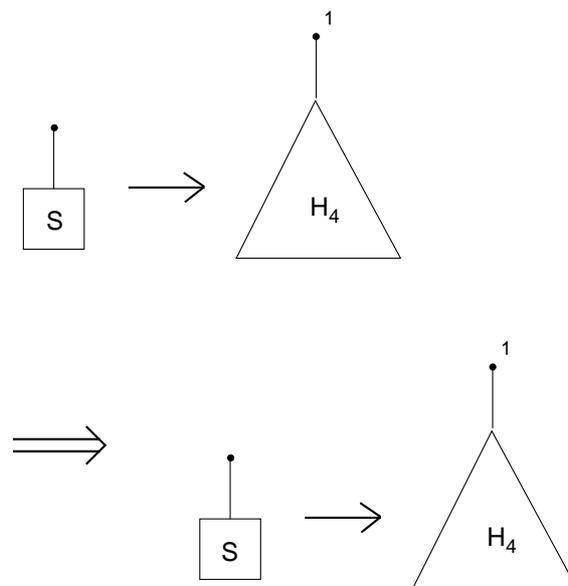


Abbildung 4.18: Ersetzungsregel 4 zur Umwandlung von untenlinear nach obenlinear

# Kapitel 5

## Kontextfreie Hypergraph-Grammatiken und L-attributierte Grammatiken

In diesem Kapitel zeigen wir, daß die Termsprachen der L-attribuierten Grammatiken und die der kontextfreien Hypergraph-Grammatiken, deren Satzformen alle Dschungel sind, identisch sind. Dazu wird in den beiden folgenden Abschnitten jeweils die Inklusion in die eine und die andere Richtung nachgewiesen.

### 5.1 Simulation von L-attribuierten Grammatiken durch kontextfreie Hypergraph-Grammatiken mit Dschungel-Eigenschaft

In diesem Abschnitt soll gezeigt werden, daß die Termsprachen L-attribuierteter Grammatiken auch Termsprachen von Hypergraph-Grammatiken sind. Dazu wird ein Konstruktionsverfahren angegeben, daß zu einer L-attribuierten Grammatik eine Hypergraph-Grammatik erzeugt. Dann wird – in mehreren Schritten – gezeigt, daß die konstruierte Hypergraph-Grammatik die gleiche Termsprache besitzt.

Eine wesentliche Idee bei der Konstruktion ist die Bijektion zwischen Attributen und Tentakeln (was eine Bijektion zwischen den Nichtterminalen impliziert).

Bevor die Konstruktion im Detail vorgestellt wird, geben wir ein Beispiel mit einer L-attribuierten Produktion einerseits und der dazu konstruierten Hypergraph-Produktion andererseits. Man mag daran bereits einen Teil der Konstruktionsideen erkennen und darauf nach der Konstruktionsbeschreibung noch einmal zurückkommen:



**Beispiel 5.1** Die gegebene L-attributierte Produktion besteht aus der kontextfreien Produktion  $A \rightarrow BC$  und den drei Berechnungsvorschriften  $Inh_B = 7 * Inh_A + 2$ ,  $Inh_C = Syn_B * Inh_B * Inh_A$  und  $Syn_A = (Syn_B + Inh_C + Syn_C) * 4$ . Die drei nichtterminalen Symbole  $A, B, C$  haben also jeweils ein inherites Attribut.

Die dazu konstruierte Hypergraph-Produktion ist dann  $(A, H)$  mit  $H$  wie in Abbildung 5.1. Die Anzahl der Attribute eines nichtterminalen Symbols  $X$  findet sich wieder im Rang von  $X$  im Hypergraph (bzw. der Rang von  $A$  bedingt die Anzahl der externen Knoten von  $H$ ). Der Ausgangsknoten (bzw. der höchste externe Knoten) entspricht dabei dem synthetischen Attribut. In der Zeichnung ist jeweils in Klammern angegeben, welcher Knoten welchem Attribut zuzuordnen ist.

Jede der Berechnungsvorschriften induziert einen Teil-Hypergraphen, was in der Abbildung 5.1 durch gleiche Schattierungen angedeutet wird.

So ist z.B. die Berechnungsvorschrift  $Inh_B = 7 * Inh_A + 2$ , die bei Linksanalyse als erste ausgewertet wird, im unteren, linken Bereich von  $H$  wiederzufinden. Da sich in der gewählten Darstellung die Eingangskante eines Knoten immer senkrecht unter diesem befindet (und alle übrigen Tentakel der Eingangskante mit weiter unten liegenden Knoten verbunden sind), müssen zur Bestimmung des assoziierten Terms eines Knotens nur von (einigen) tiefergelegenen Knoten die assoziierten Terme berechnet werden. Also ist der untere, linke Bereich von  $H$  auswertbar, ohne das dafür die zu höhergelegenen Knoten assoziierten Terme bestimmt werden müssen.

Entsprechend findet sich z.B. die Abhängigkeit, daß das synthetische Attribut (möglicherweise) erst berechenbar ist, wenn alle übrigen Attribute in dieser Produktion berechnet wurden, darin wieder, daß der Knoten  $ext(2)$  sich ganz oben befindet.

Wie bereits im Beispiel zu sehen war, werden Funktionen mit  $n$  Argumenten durch Hyperkanten mit  $n + 1$  Tentakeln dargestellt. Wir definieren daher:

**Definition 5.2** ( $inc(\Gamma)$ ) Sei  $\Gamma$  ein Marken-Alphabet. Dann ist  $inc(\Gamma) = \{X \mid X \in \Gamma\}$  ein Marken-Alphabet mit  $rank_{inc(\Gamma)}(X) = rank_{\Sigma}(X) + 1$ .

Um die eigentliche Konstruktionsbeschreibung zu entzerren, definieren wir zunächst den zu einem Term assoziierten Dschungel. Dies ist, wie der Name schon verspricht, etwa die Umkehrung zu Definition 1.27 (assoziierter Term).

**Definition 5.3** (zu einem Term assoziierter Dschungel) Sei  $\Gamma$  ein Marken-Alphabet,  $Y = (y_1, \dots, y_k)$  eine Folge von  $k$  unterschiedlichen Variablen und  $t$  ein Term über  $\Gamma \cup Y$ .

Dann ist der mit  $t$  und  $Y$  assoziierte Dschungel, kurz  $jung(t, Y)$  bezeichnet, der wie folgt induktiv definierte Dschungel vom Rang  $k + 1$  mit  $k$  Variablen über  $inc(\Gamma)$ . (Dabei entspricht  $ext(i)$  jeweils  $y_i$  für alle  $1 \leq i \leq k$ .)

1. Falls  $t = y_i$ ,  $1 \leq i \leq k$ , dann ist  $jung(t, Y) = (\{y_1, \dots, y_k\}, \emptyset, \emptyset, \emptyset, (y_1, \dots, y_k, y_i))$

2. Sei  $t = \gamma(t_1, \dots, t_n)$  mit  $\gamma \in \Gamma$ ,  $\text{rank}_\Sigma(\gamma) = n$ ,  $n \geq 0$ .

Sei  $\text{jung}(t_i, Y) = H_i$  für alle  $1 \leq i \leq n$ .

Man nehme isomorphe Kopien von  $H_1, \dots, H_n$ , so daß für alle  $i, j \in [1 : n]$  mit  $i \neq j$  gilt:

$V_{H_i} \cap V_{H_j} = \{y_1, \dots, y_k\}$  und  $E_{H_i} \cap E_{H_j} = \emptyset$ .

Dann ist  $\text{jung}(t, Y) = (V, E, \text{nod}, \text{lab}, \text{ext})$  mit

- $V = \{v\} \cup \bigcup_{i=1}^n V_{H_i}$  ( $v$  sei ein neuer Knoten:  $v \notin V_{H_i}$ ,  $1 \leq i \leq n$ )

- $E = \{e_{\text{neu}}\} \cup \bigcup_{i=1}^n E_{H_i}$  ( $e_{\text{neu}}$  sei eine neue Hyperkante)

- 

$$\text{nod}(e) = \begin{cases} (\text{ext}_{H_1}(k+1), \dots, \text{ext}_{H_n}(k+1), v) & \text{falls } e = e_{\text{neu}} \\ \text{nod}_{H_i}(e) & \text{falls } e \in E_{H_i}, 1 \leq i \leq n \end{cases}$$

- 

$$\text{lab}(e) = \begin{cases} \gamma & \text{falls } e = e_{\text{neu}} \\ \text{lab}_{H_i}(e) & \text{falls } e \in E_{H_i}, 1 \leq i \leq n \end{cases}$$

- $\text{ext} = (y_1, \dots, y_k, v)$ .

### Bemerkungen:

1. Der mit  $t$  und  $Y$  assoziierte Dschungel ist ein reiner Dschungel.
2. Der mit  $\text{jung}(t, Y)$  assoziierte Term ist  $t$ .

Kommen wir jetzt zu einem der Kernpunkte, nämlich der Konstruktion einer „passenden“ Hypergraph-Produktion bei gegebener L-attributierter Produktion:

### 5.1.1 Konstruktion

#### Konstruktionsbeschreibung:

Gegeben:

Eine L-attributierte Grammatik  $G$  mit

- $G_{CF} = (N_0, T_0, P_0, S_0)$

- $D = (\Sigma, \Lambda)$

Konstruiere Hypergraph-Grammatik  $\tilde{G} = (\tilde{\Sigma}, \tilde{\Delta}, \tilde{P}, \tilde{S})$  mit:

- $\tilde{\Sigma} = \Sigma \cup \Lambda \cup N_0$

- $\tilde{\Delta} = \Sigma \cup \Lambda$

- $\tilde{P} = \{\tilde{p} \mid p \in P_0\}$
- $\tilde{S} = S_0$
- 

$$\text{rank}_{\tilde{\Sigma}}(x) = \begin{cases} 1 & \text{falls } x \in \Sigma \\ \text{rank}_{\Lambda}(x) + 1 & \text{falls } x \in \Lambda \\ |\text{Att}(x)| + 1 & \text{falls } x \in N_0 \text{ (Anzahl der Attribute)} \end{cases}$$

$T_0$  fällt weg, da es hier um die Termsprache geht.

Bleibt noch die Beschreibung der Produktionen:

Ist  $p : B_0 \rightarrow \alpha_0 B_1 \alpha_1 \dots B_n \alpha_n$ ,  $n \geq 0$ , eine Produktion aus  $P$ , und hat  $B_i$ ,  $i \in [0 : n]$ ,

- das synthetische Attribut  $A_0^{B_i}$  und
- die inheriten Attribute  $A_1^{B_i}, \dots, A_{|\text{Inh}(B_i)|}^{B_i}$ ,  $|\text{Inh}(B_i)| \geq 0$ ,

so ist  $\tilde{p} = (B_0, \tilde{H})$  und  $\tilde{H}$  wird wie folgt konstruiert:

Sei  $Y = (y_1, \dots, y_k)$  eine Folge von  $k$  Variablen,

$$k = |\text{Inh}(B_0)| + \sum_{j=1}^n (|\text{Inh}(B_j)| + 1) + 1$$

Ordne der Variablen  $y_m$  das Attribut

$A_l^{B_0}$  zu falls  $m = l \leq |\text{Inh}(B_0)|$  (das  $l$ -te inherite von  $B_0$ )

$A_l^{B_q}$  zu falls  $m = l + |\text{Inh}(B_0)| + \sum_{j=1}^{q-1} (|\text{Inh}(B_j)| + 1)$  (das  $l$ -te inherite von  $B_q$ )

$A_0^{B_q}$  zu falls  $m = |\text{Inh}(B_0)| + \sum_{j=1}^q (|\text{Inh}(B_j)| + 1)$  (das synthetische von  $B_q$ )

$A_0^{B_0}$  zu falls  $m = k$

Es handelt sich um eine eindeutige Zuordnung.

Es sei

$$t_j^i = \begin{cases} \text{Berechnungsvorschrift für das synthetische Attribut von } B_i & \text{falls } i = j = 0 \\ \text{Berechnungsvorschrift für das } j\text{-te inherite Attribut von } B_i & j \neq 0 \wedge i \neq 0 \end{cases}$$

und  $H_j^i = \text{jung}(t_j^i, Y_j^i)$ ,  $Y_j^i = \{y_1, \dots, y_m\} \subset Y$ ,

$$m = \begin{cases} k & \text{falls } i = j = 0 \\ |\text{Inh}(B_0)| + \sum_{k=1}^{i-1} (|\text{Inh}(B_k)| + 1) + j - 1 & \text{sonst} \end{cases}$$

Ordne außerdem dem höchsten externen Knoten jeweils  $y_{m+1}$  zu.

Dann wird der Hypergraph  $\tilde{H}$  wie folgt aufgebaut:

01  $H := H_0^0$ ; (\*  $t_0^0$  ist Berechnungsvorschrift für das synthetische Attribut von  $B_0$  \*)

02  $j := n$ ;

03 Solange  $j \geq 1$  ist:

04 { (\*  $B_j$  „einsetzen“ \*)

05  $H_{neu} := (V_H, E_H \cup \{e_{neu}\}, nod, lab, ext)$  mit

$$06 \quad nod(e) = \begin{cases} nod_H(e) & \text{falls } e \in E_H \\ y_{m-p}y_{m-p+1}\dots y_m, \\ m = |Inh(B_0)| + \sum_{i=1}^j (|Inh(B_i)| + 1), \\ p = |Inh(B_j)|, & \text{falls } e = e_{neu} \end{cases}$$

$$07 \quad lab(e) = \begin{cases} lab_H(e) & \text{falls } e \in E_H \\ B_j & \text{falls } e = e_{neu} \end{cases}$$

08  $ext = y_1y_2\dots y_{m-1}y_k$ ,  $m = |Inh(B_0)| + \sum_{i=1}^j (|Inh(B_i)| + 1)$ ;

(\*  $B_j$  ist Eingangskante von  $y_m$ , deswegen wird  $y_m$  aus  $ext$  entfernt \*)

09  $H := H_{neu}$ ;

10  $l := |Inh(B_j)|$ ;

(\* Berechnungsvorschriften für die inheriten Attribute von  $B_j$  einsetzen: \*)

11 Solange  $l \geq 1$  ist:

12 {

(\* wähle isomorphe Kopien von  $H_l^j$ , so daß  $V_H \cap V_{H_l^j} = \{y_1, \dots, y_m\}$ , \*)

(\*  $m = |Inh(B_0)| + \sum_{i=1}^{j-1} (|Inh(B_i)| + 1) + l$  und  $E_H \cap E_{H_l^j} = \emptyset$  \*)

13  $H_{neu} := (V_H \cup V_{H_l^j}, E_H \cup E_{H_l^j}, nod, lab, ext)$  mit

$$14 \quad nod(e) = \begin{cases} nod_H(e) & \text{falls } e \in E_H \\ nod_{H_l^j}(e) & \text{sonst} \end{cases}$$

$$15 \quad lab(e) = \begin{cases} lab_H(e) & \text{falls } e \in E_H \\ lab_{H_l^j}(e) & \text{sonst} \end{cases}$$

16  $ext = y_1y_2\dots y_{m'+l-1}y_k$ ,  $m' = |Inh(B_0)| + \sum_{i=1}^{j-1} (|Inh(B_i)| + 1)$ ;

(\*  $ext_H$  ist  $y_1y_2\dots y_{m'+l}y_k$ ,  $ext_{H_l^j}$  ist  $y_1y_2\dots y_{m'+l}$ , bei der Vereinigung wird daher  $y_{m'+l}$  aus  $ext$  entfernt, da er eine Eingangskante aus  $E_{H_l^j}$  besitzt. \*)

17  $H := H_{neu}$ ;

18  $l := l - 1$ ;

19 } (\* Zeile 12 \*)

20  $j := j - 1$ ;

21 } (\* Zeile 04 \*)

22  $\tilde{H} := H$ ;

Bevor wir die konstruierte Hypergraph-Grammatik untersuchen, kehren wir noch

einmal zu dem Beispiel 5.1 zurück und geben die einzelnen Schritte zur Konstruktion des Hypergraphen an:

**Beispiel 5.4** Anwendung des Konstruktionsverfahren auf die L-attributierte Produktion aus dem Beispiel 5.1.

Wir haben die folgenden Zuordnungen zwischen den Attributen und den Variablen:

- $y_1$  ist  $Inh_A$  zugeordnet.
- $y_2$  ist  $Inh_B$  zugeordnet.
- $y_3$  ist  $Syn_B$  zugeordnet.
- $y_4$  ist  $Inh_C$  zugeordnet.
- $y_5$  ist  $Syn_C$  zugeordnet.
- $y_6$  ist  $Syn_A$  zugeordnet.

Jetzt wenden wir den Konstruktionsalgorithmus an, greifen jeweils die Zeilen heraus, in denen  $H$  geändert wird, und geben  $H$  als Skizze an:

1. Zeile 01,  $H := H_0^0$   
 $(t_0^0 = *(+(+(Syn_B, Inh_C), Syn_C), 4) \simeq *(+(+(y_3, y_4), y_5), 4))$   
 Siehe Abbildung 5.2.
2. Zeile 09,  $H := H_{neu}, j = 2$   
 („Einfügen“ von  $B_n = B_2 \simeq C$ )  
 Siehe Abbildung 5.3.
3. Zeile 17,  $H := H_{neu}, j = 2, l = 1$   
 („Einsetzen“ von  $H_1^2$  [ $B_2 \simeq C$ , also Berechnungsvorschrift für  $Inh_C$  ( $\simeq y_4$ ):  
 $t_1^2 = *(Syn_B, *(Inh_B, Inh_A)) \simeq *(y_3, *(y_2, y_1))$ ])  
 Siehe Abbildung 5.4.
4. Zeile 09,  $H := H_{neu}, j = 1, l = 0$   
 („Einfügen“ von  $B_1 \simeq B$ )  
 Siehe Abbildung 5.5.
5. Zeile 17,  $H := H_{neu}, j = 1, l = 1$   
 („Einsetzen“ von  $H_1^1$  [ $B_1 \simeq B$ , also Berechnungsvorschrift für  $Inh_B$  ( $\simeq y_2$ ):  
 $t_1^1 = +(*(7, Inh_A), 2) \simeq +(*(7, y_1), 2)$ ])  
 Siehe Abbildung 5.6.

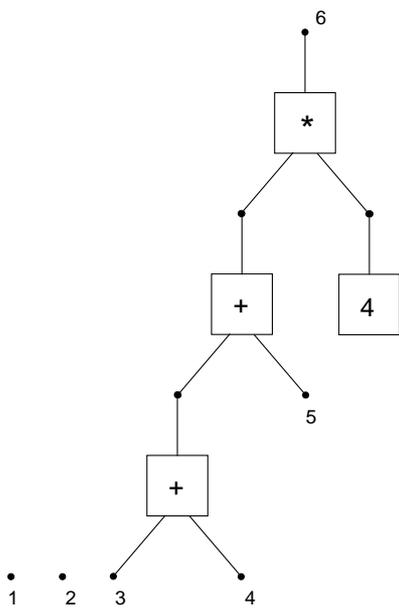


Abbildung 5.2: Schritt 1

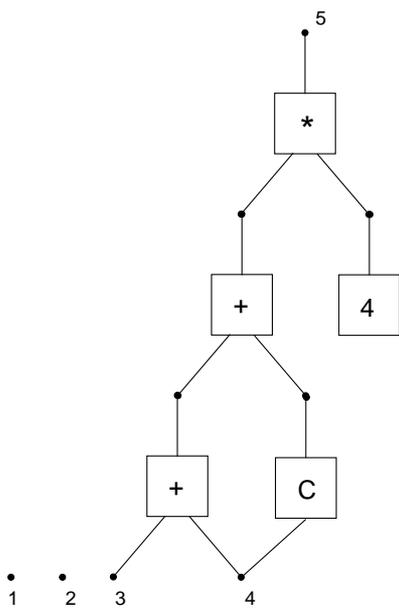


Abbildung 5.3: Schritt 2

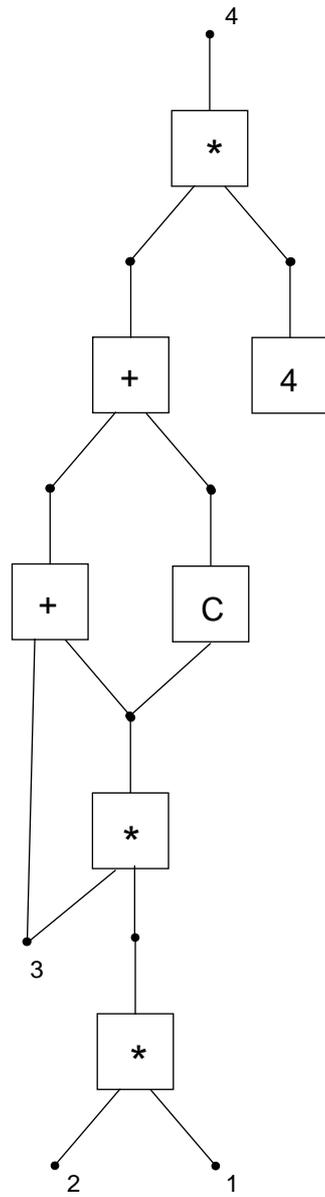


Abbildung 5.4: Schritt 3

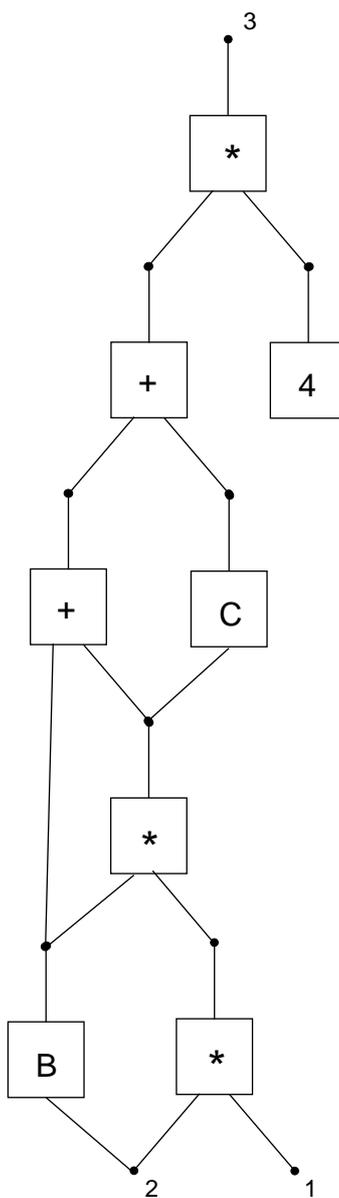


Abbildung 5.5: Schritt 4

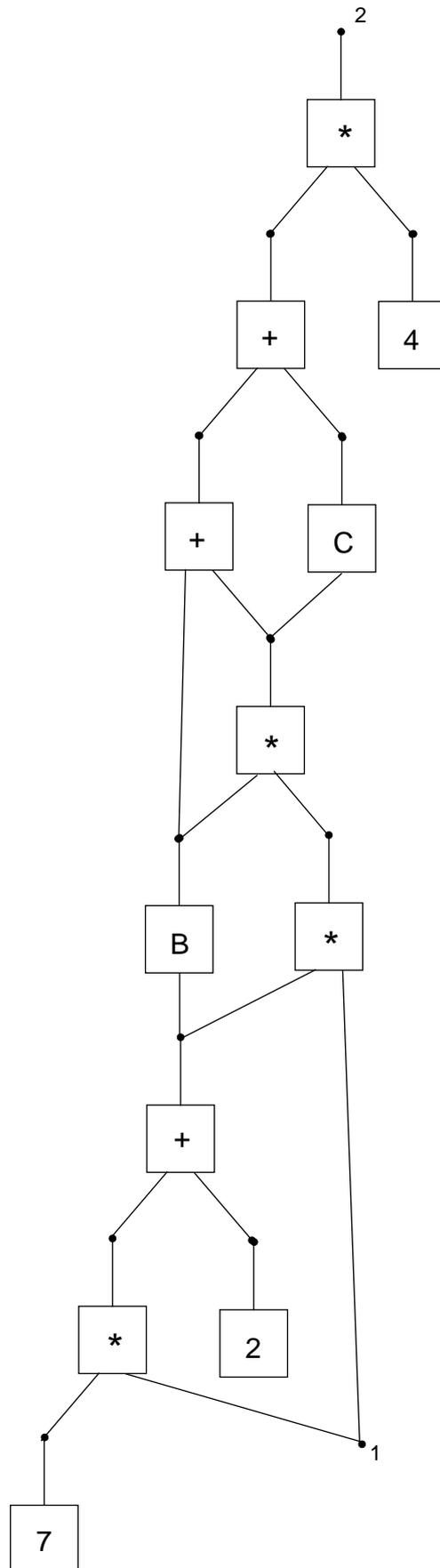


Abbildung 5.6: Schritt 5

### 5.1.2 Vorbereitungen

Wir zeigen zunächst, daß die konstruierte Hypergraph-Grammatik die Dschungel-Eigenschaft besitzt, und dann, daß sie die „richtigen“ Terme erzeugt.

**Behauptung:**  $\tilde{H}$  ist ein  $(|Inh(B_0)| + 1)$ -Dschungel mit  $|Inh(B_0)|$  Variablen.

**Beweis:**

Alle  $H_l^j$  ( $j \in [1 : \Theta_i]$ ,  $l \in [1 : |Inh(B_j)|]$ ) sind Dschungel mit Variablen.

Beh.:  $H$  ist ein Dschungel mit Variablen.

I.A.:  $H = H_0^0 \checkmark$

I.S.:

1.  $H_{neu}$  „entsteht“ aus  $H$  und  $H_l^j$ , dabei gilt

$$|ext_H| = |ext_{H_l^j}| + 1, \text{ genauer:}$$

wenn  $ext_H = (y_1, \dots, y_{s-1}, y_s, y_k)$  ist, dann ist

$ext_{H_l^j} = (y_1, \dots, y_{s-1}, y_s)$  und

$ext_{H_{neu}} = (y_1, \dots, y_{s-1}, y_k)$

Die internen Knoten von  $H_{neu}$  sind one-incoming:

- entweder aus  $H$ ,  $H$  ist 1-Dschungel (mit  $s$  Variablen)
- oder aus  $H_l^j$ ,  $H_l^j$  ist 1-Dschungel (mit  $s - 1$  Variablen)
- oder  $y_s$ , Eingangskante „aus“  $H_l^j$

Annahme,  $H_{neu}$  sei nicht zyklensfrei:

- Zyklus in  $H$  ? Widerspruch !
- Zyklus in  $H_l^j$  ? Widerspruch !
- Zyklus in  $H$  und  $H_l^j$  ? Übergang von  $H$  nach  $H_l^j$  unmöglich: Widerspruch !

□

Die anderen Eigenschaften sind offensichtlich erfüllt.

2.  $H_{neu}$  „entsteht“ durch Einfügen von  $B_j$ , dann ... (entsprechende Argumentation).

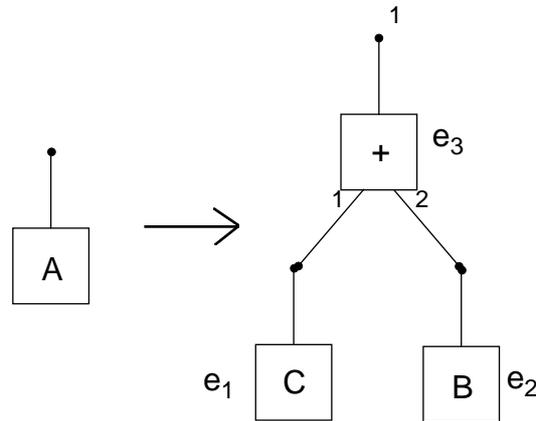
□

**Behauptung:** Die konstruierte Hypergraph-Grammatik  $\tilde{G} = (\tilde{\Sigma}, \tilde{\Delta}, \tilde{P}, \tilde{S})$  besitzt die Dschungel-Eigenschaft.

**Beweis:**

Sei  $p = (B_0, \tilde{H})$  eine Produktion aus  $\tilde{G}$ , dann gilt

- a)  $\tilde{H}$  ist zyklensfrei, da Dschungel mit Variablen zyklensfrei sind.
- b)  $rank_{\tilde{\Sigma}}(x) \geq 1 \checkmark$
- c)  $rank_{\tilde{\Sigma}}(\tilde{S}) = (|Inh(S_0)| + 1) = 0 + 1 = 1 \checkmark$
- d) siehe vorhergehende Behauptung.

Abbildung 5.7: Zu  $A \rightarrow BC$  konstruierte Produktion

□

**Bemerkung 1:** Die konstruierte Hypergraph-Grammatik  $\tilde{G}$  muß nicht die reiner-Dschungel-Eigenschaft besitzen; es kann durchaus eine Produktion  $\tilde{p}_i = (B_0, H_i)$  in  $\tilde{P}$  geben, bei der für eine Hyperkante aus  $H_i$  gilt, daß von ihr aus kein Pfad zum höchsten externen Knoten von  $H_i$  führt.

Wenn beispielsweise  $B_0 \rightarrow \alpha_0 B_1 \alpha_1 \dots B_{\Theta_i} \alpha_{\Theta_i}$  die zur Konstruktion herangezogene L-attributierte Produktion ist und das synthetische Attribut von  $B_k$  in keiner Berechnungsvorschrift „verwendet“ wird, tritt dieser Fall ein.

**Bemerkung 2:** Wenn  $\tilde{G}$  reiner-Dschungel-Eigenschaft besitzt, so muß die – nur dann definierte – *vl*-Ordnung der nichtterminalen Symbole der rechten Seite der zu  $p_i : B_0 \rightarrow \alpha_0 B_1 \alpha_1 \dots B_{\Theta_i} \alpha_{\Theta_i}$  konstruierten Produktion  $\tilde{p}_i = (B_0, H_i)$  nicht der Ordnung in  $p_i$  entsprechen ( $B_k <_{vl} B_m$  mit  $1 \leq m \leq k \leq \Theta_i$  ist möglich).

Hierzu ein kleines Beispiel:

**Beispiel 5.5** Sei  $A \rightarrow BC$  eine L-attributierte Produktion mit der Berechnungsvorschrift  $Syn_A = +(Syn_C, Syn_B)$  (die nichtterminalen Symbole haben jeweils nur ein Attribut). Dann wird dazu die Produktion aus Abbildung 5.7 konstruiert und es gilt  $ord_N = BC$ , aber  $e_1 <_{vl} e_2$ .

### 5.1.3 Ableitungsteilbäume

**Definition 5.6 (Ableitungsteilbaum)** Sei  $G$  eine Hypergraph-/ Makro-/ L-attributierte Grammatik mit dem Startsymbol  $S$  und sei für jede Produktion  $p$  aus  $G$  eine totale Ordnung zwischen den nichtterminalen Symbolen der rechten Seite von  $p$  festgelegt.

Der markierte Baum  $t$  heißt genau dann Ableitungsteilbaum zu der Grammatik  $G$ , wenn gilt:

1) Die Anzahl der Nachfolger eines mit  $p$  markierten Knotens entspricht der Anzahl der nichtterminalen Symbole der rechten Seite von  $p$ .

2) Ist  $k_i$  der mit  $p_i$  markierte  $i$ -te Nachfolger des mit  $p$  markierten Knotens  $k$  und sind  $B_1, \dots, B_m$ ,  $m \geq 0$ , die nichtterminalen Symbole der rechten Seite von  $p$ , wobei diese Aufzählung die totale Ordnung wiedergibt, dann steht  $B_i$  auf der linken Seite von  $p_i$ .

Der Ableitungsteilbaum  $t$  heißt auch Ableitungsbaum, wenn außerdem gilt:

3) Die Wurzel von  $t$  ist mit einer Produktion markiert, deren linke Seite  $S$  ist.

### Bemerkungen:

1. Genau die Blätter sind mit terminalen Produktionen markiert.
2. Im Falle einer kontextfreien oder (L-)attributierten Grammatik wird als totale Ordnung zwischen den nichtterminalen Symbolen der rechten Seite einer Produktion typischerweise die Reihenfolge ihres Auftretens von links nach rechts gewählt.
3. Im Falle einer Hypergraph-Grammatik mit reiner-Dschungel-Eigenschaft bietet sich die  $vl_N$ -Ordnung an.
4. Im Falle einer IO-Makrogrammatik sollten „innere“ nichtterminale Symbole vor „äußeren“ angeordnet werden (damit die Produktionen von links nach rechts abgearbeitet werden können).

### Vereinbarung:

Sofern nicht anders angegeben, ist die für den Ableitungsteilbaum einer kontextfreien / L-attributierten Grammatik verwendete Ordnung zwischen den nichtterminalen Symbolen der rechten Seiten der Produktionen die Reihenfolge ihres Auftretens von links nach rechts.

Es lassen sich einige Zusammenhänge zwischen Ableitungen und Ableitungsteilbäumen feststellen:

1. Eine terminale Ableitung (d.h., nach dem letzten Ableitungsschritt enthält das abgeleitete Wort keine nichtterminalen Symbole mehr) legt eindeutig einen Ableitungsteilbaum fest.
2. Für kontextfreie und (L-)attributierte Grammatiken legt ein Ableitungsteilbaum eineindeutig eine Linksableitung fest.
3. Ein Ableitungsteilbaum legt eindeutig ein terminales Wort fest.

4. Eine terminale Linksableitung beginnend bei einem einzelnen nichtterminalen Symbol  $B$  einer L-attribuierten Grammatik legt (*per definitionem*) einen Term  $t$  zur Berechnung des Wertes des synthetischen Attributes von  $B$  fest – in  $t$  tauchen gegebenenfalls die inheriten Attribute von  $B$  als Argumente auf.
5. Also legt ein Ableitungsteilbaum zu einer L-attribuierten Grammatik für das synthetische Attribut von  $B$  – es sei  $B$  die linke Seite der die Wurzel markierenden Produktion – einen Term  $t$  (wie oben) fest.
6. Speziell legt ein Ableitungsbaum einen Term zur Berechnung des designierten Attributes fest.

**Definition 5.7** Sei  $G$  eine L-attribuierte Grammatik und  $\tilde{G}$  die dazu konstruierte Hypergraph-Grammatik.

Sei  $p : B_0 \rightarrow \alpha_0 B_1 \alpha_1 \dots B_n \alpha_n$ ,  $n \geq 0$  eine L-attribuierte Produktion aus  $G$  und  $\tilde{p} = (B_0, \tilde{H})$  die dazu konstruierte Hypergraph-Produktion.

Sei  $e_i$  die zu  $B_i$  konstruierte Hyperkante (Zeile 04 bis 09) für alle  $i \in [1 : n]$ .

Dann ist  $\text{ord}_N(\tilde{H}) = e_1 \dots e_n$  die  $\tilde{H}$  durch die Konstruktion zugeordnete Ordnung der Hyperkanten mit nichtterminalen Label von  $\tilde{H}$ .

**Vereinbarung:**

Sofern nicht anders angegeben, ist die für den Ableitungsteilbaum einer Hypergraph-Grammatik verwendete Ordnung zwischen den nichtterminalen Symbolen der rechten Seiten der Produktionen  $\text{ord}_N$ .

**Definition 5.8** Sei  $G$  eine L-attribuierte Grammatik mit der Produktionsmenge  $P$  und  $\tilde{G}$  die zu  $G$  konstruierte Hypergraph-Grammatik mit der Produktionsmenge  $\tilde{P}$ . Dann sei

$f_P : P \rightarrow \tilde{P}$  mit  $f_P(p) = \tilde{p} \Leftrightarrow \tilde{p}$  ist die zu  $p$  konstruierte Hypergraph Produktion.

Offensichtlich ist  $f_P$  eine bijektive Abbildung zwischen den Produktionen der Grammatiken  $G$  und  $\tilde{G}$ . Sie läßt sich auf Ableitungsteilbäume bzw. markierte Bäume erweitern:

**Definition 5.9** Sei  $G$  eine L-attribuierte Grammatik mit der Produktionsmenge  $P$  und  $\tilde{G}$  die zu  $G$  konstruierte Hypergraph-Grammatik mit der Produktionsmenge  $\tilde{P}$ .

Sei  $t$  ein markierter Baum mit Marken aus  $P$ .

Dann sei  $f(t)$  der markierte Baum, den man erhält, wenn man auf jede Marke von  $t$  die Abbildung  $f_P$  anwendet (jede Marke  $p$  in  $t$  durch die Marke  $f_P(p)$  ersetzt).

Die Motivation für die obigen Definitionen liegt in folgendem Satz:

**Satz 5.10** *Sei  $G$  eine  $L$ -attributierte Grammatik,  $\tilde{G}$  die zu  $G$  konstruierte Hypergraph-Grammatik und sei  $t$  ein Ableitungsteilbaum zu  $G$ . Dann ist  $f(t)$  ein Ableitungsteilbaum zu  $\tilde{G}$ .*

**Beweis:** Induktion über die Höhe  $h$  von  $t$ .

$h = 1$ :

Sei  $p$  die Marke an der Wurzel  $k$  von  $t$ , da  $k$  keine Nachfolger hat, muß  $p$  eine terminale Produktion sein, deswegen muß auch  $f_P(p)$  eine terminale Produktion sein.  $f(t)$  besitzt als Wurzelmarkierung die terminale Produktion  $f_P(p)$ , ist also Ableitungsteilbaum zu  $\tilde{G}$ .

$h \rightarrow h + 1$ :

Sei  $p : B_0 \rightarrow \alpha_0 B_1 \alpha_1 \dots B_n \alpha_n$  die Marke an der Wurzel  $k$  von  $t$ . Da  $t$  Ableitungsteilbaum zu  $G$  ist, folgt, daß  $k$   $n$  Nachfolgerknoten besitzt. Seien  $k_1, \dots, k_n$  die Nachfolger von  $k$  und  $p_1, \dots, p_n$  die Marken an diesen Knoten. Dann gilt:

$lhs(p_i) = B_i$ .

Sei  $\tilde{k}$  das Bild von  $k$ , also der mit  $f_P(p)$  markierte Wurzelknoten von  $f(t)$ . Sei  $\tilde{k}_i$ ,  $1 \leq i \leq n$ , das Bild von  $k_i$ , also der mit  $f_P(p_i)$  markierte  $i$ -te Nachfolger von  $\tilde{k}$ . Aus der Konstruktion folgt, daß

1.  $lhs(f_P(p)) = B_0$ ,
2.  $lab(ord_N(rhs(f_P(p)))) = B_1 \dots B_n$  (vergleiche Definition 5.7) und
3.  $lhs(f_P(p_i)) = B_i$  ist.

Also erfüllt der Wurzelknoten  $\tilde{k}$  die Bedingungen, um Knoten eines Ableitungsteilbaums zur Grammatik  $\tilde{G}$  zu sein.

Für die übrigen Knoten von  $f(t)$  kann dies per Induktionsannahme vorausgesetzt werden.

Also gilt die Aussage auch für Bäume der Höhe  $h + 1$  und es folgt der Satz. □

Argumentiert man gerade umgekehrt, so erhält man entsprechend

**Folgerung 5.11** *Sei  $G$  eine  $L$ -attributierte Grammatik,  $\tilde{G}$  die zu  $G$  konstruierte Hypergraph-Grammatik und sei  $t$  ein Ableitungsteilbaum zu  $\tilde{G}$ . Dann ist  $f^{-1}(t)$  ein Ableitungsteilbaum zu  $G$ .*

Es existiert also eine bijektive Abbildung, die die Mengen der Ableitungsbäume aufeinander abbildet. Nun muß gezeigt werden, daß die sich entsprechenden Ableitungen auch jeweils den gleichen Term festlegen.

### 5.1.4 Nachweis der Gleichheit

**Satz 5.12** Sei  $t$  Ableitungsbaum zur  $L$ -attributierten Grammatik  $G$ ,  $s$  der durch  $t$  festgelegte Term, dessen Auswertung den Wert des designierten Attributs ergibt und  $H$  der Hypergraph, der durch  $f(t)$  festgelegt ist. Dann ist  $s$  der zu  $H$  assoziierte Term.

**Beweis:** Wir zeigen:

Ist bzw. sind

- $t$  Ableitungsteilbaum zur  $L$ -attributierten Grammatik  $G$ ,
- $p : B_0 \rightarrow \alpha_0 B_1 \alpha_1 \dots B_n \alpha_n$  die Wurzelmarkierung von  $t$ ,
- $s$  der Term zur Berechnung des synthetischen Attributs von  $B_0$ , der durch  $t$  festgelegt wird,
- $Y = \{y_1, \dots, y_{|Inh(B_0)|}\}$  eine Menge von Variablen,
- $A_1^{B_0}, \dots, A_{|Inh(B_0)|}^{B_0}$  die inheriten Attribute von  $B_0$ ,
- $s_Y$  der Term, den man erhält, wenn man in  $s$  jedes  $A_i^{B_0}$  durch  $y_i$  ersetzt ( $1 \leq i \leq |Inh(B_0)|$ ) und
- $H$  der durch  $f(t)$  festgelegte Hypergraph,

dann wird mit  $H$  der Term  $s_Y$  assoziiert.

Induktion über die Höhe  $h$  des Ableitungsteilbaums.

$h = 1$ : (Die Wurzelmarkierung muß also eine terminale Produktion sein)

Sei bzw. seien

- $p : B_0 \rightarrow \alpha_0$  Wurzelmarkierung,
- $A_0^{B_0}$  das synthetische Attribut von  $B_0$ ,
- $A_1^{B_0}, \dots, A_{|Inh(B_0)|}^{B_0}$  die inheriten Attribute von  $B_0$ ,
- $s_0^0$  die Berechnungsvorschrift für  $A_0^{B_0}$  und
- $s_Y$  der Term den man erhält, indem man in  $s_0^0$  jedes  $A_i^{B_0}$  durch  $y_i$  ( $1 \leq i \leq |Inh(B_0)|$ ) ersetzt.

In der Konstruktion der Hypergraph-Produktion  $f_P(p)$  werden im wesentlichen die Zeilen

01 ( $H := H_0^0$ ) und

22 ( $\tilde{H} := H$ )

ausgeführt,  $\tilde{H}$  also gerade auf  $\text{jung}(s_0^0, Y)$  gesetzt, daher wird mit  $\tilde{H}$  der Term  $s_Y$  assoziiert.

$h \rightarrow h + 1$ :

Sei bzw. seien

- $p : B_0 \rightarrow \alpha_0 B_0 \alpha_1 \dots B_n \alpha_n$  die Markierung der Wurzel  $k_0$ ,
- $k_1, \dots, k_n$  die Nachfolgerknoten von  $k_0$  (von links nach rechts) und
- $t_i$ ,  $1 \leq i \leq n$ , der Teil des Ableitungsteilbaums mit  $k_i$  als Wurzel.
- $A_1^{B_0}, \dots, A_{|\text{Inh}(B_0)|}^{B_0}$  die inheriten Attribute von  $B_0$  und
- $A_0^{B_0}$  das synthetische.
- $s_0^0$  die Berechnungsvorschrift für  $A_0^{B_0}$ .
- $\tilde{k}_0$  der Wurzelknoten von  $f(t)$ ,
- $\tilde{k}_1, \dots, \tilde{k}_n$  die Nachfolger von  $\tilde{k}_0$  und
- $\tilde{t}_i$  der Teil des Ableitungsteilbaums mit  $\tilde{k}_i$  als Wurzel.
- Sei  $f_P(p) = (B_0, \tilde{H})$  die Markierung von  $\tilde{k}_0$  (also die zu  $p$  konstruierte Hypergraph-Produktion) und  $h_0^0$  der zu  $\tilde{H}$  assoziierte Term.

$s_0^0$  und  $h_0^0$  unterscheiden sich (möglicherweise), können aber folgendermaßen „ange-glichen“ werden:

1. Ersetze in  $h_0^0$  jedes  $B_i(t_1, \dots, t_{|\text{Inh}(B_i)|})$  durch  $A_0^{B_i}$  (synthetisches Attribut von  $B_i$ ).
2. Ersetze in  $s_0^0$  jedes  $A_i^{B_j}$ ,  $1 \leq i \leq |\text{Inh}(B_i)|$ ,  $1 \leq j \leq n$ , durch die in  $p$  vereinbarte Berechnungsvorschrift.
3. Ersetze in  $s_0^0$  jedes  $A_i^{B_0}$ ,  $1 \leq i \leq |\text{Inh}(B_0)|$ , durch  $y_i$ .

Jetzt werden nacheinander die durch die Ableitungsteilbäume  $t_n, \dots, t_1$  (bzw.  $\tilde{t}_n, \dots, \tilde{t}_1$ ) festgelegten Terme eingefügt (die Einsetzung erfolgt „rückwärts“ ( $n, \dots, 1$ ), weil sonst die „Angleichung“ der Terme (s. o.) umständlicher ist):

$t_n$ :

- Sei  $s_0^n$  der Term, der dem synthetischen Attribut von  $B_n$  durch  $t_n$  zugeordnet wird.
- Seien  $s_i^n$ ,  $1 \leq i \leq |Inh(B_n)|$ , die den inheriten Attributen von  $B_n$  zugeordneten Terme (in  $p$  als Berechnungsvorschriften gegeben).

Dann sei  $s^n$  der Term, der durch Einsetzen der „Zwischenergebnisse“  $s_i^n$ ,  $0 \leq i \leq |Inh(B_n)|$ , entsteht:

Ersetze in  $s_0^n$  jedes Auftreten von  $A_0^{B_n}$  durch  $s_0^n$ ,  
dann ersetze jedes Auftreten von  $A_{|Inh(B_n)|}^{B_n}$  durch  $s_{|Inh(B_n)|}^n$ ,

⋮

dann ersetze jedes Auftreten von  $A_1^{B_n}$  durch  $s_1^n$ .

Sei  $s^n$  der so entstandene Term.

$\tilde{t}_n$ :

Sei  $\tilde{H}_n$  der durch  $\tilde{t}_n$  festgelegte Hypergraph (Dschungel mit  $|Inh(B_n)|$  Variablen).  
Sei  $H_n$  der Dschungel, der entsteht, wenn in  $\tilde{H}$  die Hyperkante  $B_n$  durch  $\tilde{H}_n$  ersetzt wird und sei  $h_n$  der zu  $H_n$  assoziierte Term.

Über die Induktionsannahme kann geschlossen werden, daß  $s^n$  und  $h_n$  sich „angleichen“ (s. o.) lassen.

Dann wird  $t_{n-1}$  (bzw.  $\tilde{t}_{n-1}$ ) verwendet, um  $s^n$  (bzw.  $H_n/h_n$ ) zu manipulieren, bis schließlich die Terme  $s^1$  und  $h_1$  vorliegen.

Diese beiden Terme können wieder „angeglichen“ werden, tatsächlich können aber die ersten beiden Schritte der „Angleichung“ entfallen, da es dort nichts mehr zu ersetzen gibt.

Der dritte Schritt ist allerdings die schon im Satz vorgesehene Manipulation, und da außerdem  $s^1$  bzw.  $h_1$  die durch  $t$  bzw.  $f(t)$  festgelegten Terme sind, folgt der Satz. □

**Satz 5.13** *Sei  $G$  eine L-attributierte Grammatik und  $\tilde{G}$  die dazu konstruierte Hypergraph-Grammatik.*

*Dann gilt :  $T_{LaG}(G) = T_{HG}(\tilde{G})$ .*

**Beweis:**

„ $\subseteq$ “:  $s \in T_{LaG}(G)$

$\Rightarrow$  es existiert ein Ableitungsteilbaum  $t$  zu  $G$ , der  $s$  zur Berechnung des designierten Attributs festlegt.

$\Rightarrow$  (mit vorherigen Satz)

mit dem durch  $f(t)$  festgelegten Hypergraphen wird der Term  $s$  assoziiert.

„ $\supseteq$ “:  $s \in T_{HG}(\tilde{G})$

$\Rightarrow$  es existiert ein Ableitungsteilbaum  $t$  zu  $\tilde{G}$ , der einen Hypergraphen  $H$  festlegt, zu dem der Term  $s$  assoziiert wird.

$\Rightarrow$  der markierte Baum  $f^{-1}(t)$  ist Ableitungsbaum zu  $G$ , dieser legt Term  $s'$  zur Berechnung des designierten Attributs fest.

Mit dem vorherigen Satz folgt  $s' = s$ .

Also gilt  $T_{LaG}(G) \subseteq T_{HG}(\tilde{G})$  und  $T_{LaG}(G) \supseteq T_{HG}(\tilde{G})$  und somit  $T_{LaG}(G) = T_{HG}(\tilde{G})$ . □

Aus dem Satz 5.13 folgt dann unmittelbar:

**Folgerung 5.14**  $\mathcal{T}_{LaG} \subseteq \mathcal{T}_{HG}$

## 5.2 Simulation von Hypergraph-Grammatiken mit Dschungel-Eigenschaft durch L-attributierte Grammatiken

In diesem Abschnitt geht es darum zu zeigen, daß die Menge der Termsprachen der Hypergraph-Grammatiken mit Dschungel-Eigenschaft Teilmenge der Menge der Termsprachen L-attributierter Grammatiken ist.

Dies geschieht konstruktiv, d. h. es wird zu einer vorgelegten Hypergraph-Grammatik  $G$  eine L-attributierte Grammatik  $\tilde{G}$  angegeben, die die gleiche Termsprache besitzt.

Also muß zunächst definiert werden, wie die L-attributierte Grammatik  $\tilde{G}$  zu konstruieren ist. Ähnlich wie im umgekehrten Fall wird dabei jeder nichtterminalen Hyperkante eineindeutig ein nichtterminales Symbol der L-attributierten Grammatik zugeordnet und jedes Tentakel einer nichtterminalen Hyperkante wird eineindeutig durch ein Attribut des entsprechenden nichtterminalen Symbols repräsentiert. Dieser enge Zusammenhang wird dann wieder genutzt, um zu zeigen, daß bei „gleicher“ Ableitung in den beiden Grammatiken  $G$  und  $\tilde{G}$  auch der gleiche Term festgelegt wird.

### 5.2.1 Konstruktion

**Konstruktionsbeschreibung:**

Gegeben eine kontextfreie Hypergraph-Grammatik  $G = (\Sigma, \Delta, P, S)$  mit Dschungel-

Eigenschaft.

Konstruiere die L-attributierte Grammatik  $\tilde{G}$  mit:

- $G_{cf} = (N_0, T_0, P_0, S_0)$  mit
  - $N_0 = \Sigma - \Delta$
  - $T_0 = \emptyset$  (denn wir sind nur an den Termen interessiert)
  - $P_0$  siehe unten
  - $S_0 = S$
- $D = (\tilde{\Sigma}, \Lambda)$  mit
  - $\tilde{\Sigma} = \{x \in \Delta \mid \text{rank}_\Sigma(x) > 1\}$
  - $\Lambda = \{x \in \Delta \mid \text{rank}_\Sigma(x) = 1\}$
  - $\text{rank}_{\tilde{\Sigma}}(x) = \text{rank}_\Sigma(x) - 1$
- –  $\text{Syn}(B) = \{A_0^B\}$ , (für  $B \in N_0 = \Sigma - \Delta$ )
- $\text{Inh}(B) = \{A_1^B, \dots, A_n^B\}$ ,  $n = \text{rank}_\Sigma(B) - 1 \geq 0$

Bleibt die Beschreibung der Produktionen und der Berechnungsvorschriften.

Sei  $p = (B_0, H)$  eine Hypergraph-Produktion aus  $G$  und  $e_1, \dots, e_n$  eine Aufzählung aller Hyperkanten mit nichtterminalen Label ( $\text{lab}(e_i) \in \Sigma - \Delta$ ), die die  $v$ -Ordnung einbettet (also für alle  $1 \leq i < j \leq n$  gilt:  $e_j \not\prec_v e_i$  — sprich  $e_j$  ist nicht Vorgänger von  $e_i$ ).

Sei  $\text{lab}(e_i) = B_i$  für  $i \in [1 : n]$ .

Um später auf diese Aufzählung Bezug nehmen zu können, vereinbaren wir:  $\text{ord}_N(H) = e_1 \dots e_n$  (in Anlehnung an Definition 5.7, denn auch in dieser Richtung muß eine „durch die Konstruktion“ festgelegte Ordnung berücksichtigt werden – würde man am Ende auf die konstruierte L-attributierte Grammatik  $\tilde{G}$  die Konstruktion aus dem vorhergehenden Abschnitt anwenden, würde sich wiederum die gleiche Ordnung aus der Konstruktion ergeben).

Dann ist  $\tilde{p} : B_0 \rightarrow B_1 \dots B_n$ .

Zur Angabe der Berechnungsvorschriften benötigen wir eine Abwandlung der Definition des assoziierten Terms. Für die Knoten, die eine Kante mit nichtterminalem Label als Eingangskante besitzen, wird die Festlegung des assoziierten Terms dahingehend geändert, daß diesem Knoten das „synthetische Attribut“ als assoziierter Term zugeordnet wird. Den ersten  $\text{rank}(H) - 1$  externen Knoten werden die inneren Attribute von  $B_0$  zugeordnet.

(Erinnerung:  $p = (B_0, H)$ ,  $ord_N(H) = e_1 \dots e_n$ )

$$Term_N(v, H) = \begin{cases} A_j^{B_0} & \text{falls } v = ext(j), j < rank_\Sigma(B_0) \\ A_0^{B_i} & \text{falls } e_i \text{ Eingangs-Kante von } v \\ & \text{und } lab(e_i) = B_i \in \Sigma - \Delta \\ lab(e) & \text{falls } e \text{ Eingangs-Kante von } v, \\ & lab(e) \in \Delta \text{ und } Rank_\Sigma(e) = 1 \\ lab(e)(t_1, \dots, t_n) & \text{falls } e \text{ Eingangs-Kante von } v, \\ & lab(e) \in \Delta, Rank_\Sigma(e) = n + 1 > 1 \\ & \text{und } t_j = Term_N(nod(e, j), H), 1 \leq j \leq n \end{cases}$$

$$Term_N(H) = Term_N(ext(rank(H)), H)$$

Für die inheriten Attribute von  $B_i$  ( $1 \leq i \leq n$ ) werden die folgenden Berechnungsvorschriften festgelegt:

$$A_j^{B_i} = Term_N(v, H) \text{ mit } v = nod(e_i, j) \text{ und } j \in [1 : rank_\Sigma(B_i) - 1]$$

und für das synthetische Attribut von  $B_0$ :

$$A_0^{B_0} = Term_N(ext(rank_\Sigma(B_0)), H) = Term_N(H) \quad (\text{es ist } rank_\Sigma(B_0) = rank(H))$$

Damit ist die Grammatik  $\tilde{G}$  vollständig beschrieben.

## 5.2.2 Vorbereitungen

Zunächst ist zu zeigen, daß diese attributierte Grammatik eine L-attributierte Grammatik ist.

**Satz 5.15**  $\tilde{G}$  ist eine L-attributierte Grammatik.

**Beweis:**

Annahme, daß die attributierte Grammatik  $\tilde{G}$  nicht L-attributiert ist.

D. h. es gibt eine Produktion  $\tilde{p} : B_0 \rightarrow B_1 \dots B_n$  mit einer Berechnungsvorschrift für das Attribut  $A_j^{B_i}$  ( $1 \leq i \leq n, 1 \leq j \leq |Inh(B_j)|$ ), bei der ein Attribut  $A_l^{B_k}$  ( $i \leq k \leq n, 0 \leq l \leq |Inh(B_k)|$ ) oder das Attribut  $A_0^{B_0}$  verwendet wird.

$A_0^{B_0}$  taucht in  $Term_N(v, H)$  nicht auf, bleibt  $A_l^{B_k}$ :

Da in der Definition von  $Term_N$  nur „synthetische“ Attribute verwendet werden, folgt  $l = 0$ . Ferner kann aus der Definition von  $Term_N$  geschlossen werden, daß es dann einen Pfad von  $nod(e_k, rank_\Sigma(B_k))$  zu  $nod(e_i, j)$  gibt, also ist  $e_k$  Vorgänger von  $e_i$  – Widerspruch!

□

In Umkehrung der Definitionen 5.8 und 5.9 definieren wir

**Definition 5.16** Sei  $G$  eine Hypergraph-Grammatik mit der Produktionenmenge  $P$  und  $\tilde{G}$  die zu  $G$  konstruierte L-attributierte Grammatik mit der Produktionenmenge  $\tilde{P}$ . Dann sei

$g_P : P \rightarrow \tilde{P}$  mit  $g_P(p) = \tilde{p} \Leftrightarrow \tilde{p}$  ist die zu  $p$  konstruierte L-attributierte Produktion.

und

**Definition 5.17** Sei  $G$  eine Hypergraph-Grammatik mit der Produktionenmenge  $P$  und  $\tilde{G}$  die zu  $G$  konstruierte L-attributierte Grammatik mit der Produktionenmenge  $\tilde{P}$ .

Sei  $t$  ein markierter Baum mit Marken aus  $P$ .

Dann sei  $g(t)$  der markierte Baum, den man erhält, wenn man auf jede Marke von  $t$  die Abbildung  $g_P$  anwendet (jede Marke  $p$  in  $t$  durch die Marke  $g_P(p)$  ersetzt).

Es gilt

**Satz 5.18** (Bezeichnungen wie in obiger Definition)

$t$  ist Ableitungsteilbaum zu  $G \Leftrightarrow g(t)$  ist Ableitungsteilbaum zu  $\tilde{G}$ .

**Beweisidee:**

Die Konstruktion liefert eine Bijektion zwischen den Produktionen, außerdem eine zwischen den nichtterminalen Zeichen der rechten (und linken) Seiten der einander zuzuordnenden Produktionen. Daher kann der Beweis analog zu dem aus Satz 5.10 (und Folgerung 5.11) geführt werden.

□

### 5.2.3 Nachweis der Gleichheit

Bei „gleichem“ Ableitungsbaum wird der gleiche Term festgelegt.

**Satz 5.19** Sei  $t$  Ableitungsbaum zur Hypergraph-Grammatik  $G$ ,

$H_t$  der durch  $t$  festgelegte Dschungel und

$s$  der zu  $H_t$  assoziierte Term ( $s = \text{Term}(H_t)$ ).

Sei  $\tilde{G}$  die zu  $G$  konstruierte L-attributierte Grammatik.

Dann legt  $g(t)$  – Ableitungsbaum zur L-attributierten Grammatik  $\tilde{G}$  – gerade  $s$  als Term zur Berechnung des designierten Attributs von  $\tilde{G}$  fest.

**Beweis:** Wir zeigen:

Ist bzw. sind

- $t$  Ableitungsteilbaum zur Hypergraph-Grammatik  $G$ ,
- $p = (B_0, H)$  die Wurzelmarkierung von  $t$ ,
- $ord_N(H) = e_1 \dots e_n$ ,  $n \geq 0$ , mit  $lab(e_i) = B_i$ ,  $i \in [1 : n]$  (die zur Konstruktion von  $\tilde{G}$  verwendete Aufzählung aller nichtterminalen Hyperkanten von  $H$ ),
- $H_t$  der durch  $t$  festgelegte Hypergraph (Dschungel mit  $rank_\Sigma(B_0) - 1$  Variablen) und
- $s$  der Term zur Berechnung des synthetischen Attributs von  $B_0$ , der durch  $g(t)$  festgelegt wird.

Dann gilt  $Term_N(H_t) = s$ .

Induktion über die Höhe  $h$  des Ableitungsteilbaums.

$h = 1$ : (Die Wurzelmarkierung muß also eine terminale Produktion sein)

Sei  $p = (B_0, H)$  die Wurzelmarkierung von  $t$ , dann ist

$H_t = H$  und

$Term_N(H_t) = Term_N(H) = Term_N(ext(rank_\Sigma(B_0)), H)$ .

Die Wurzelmarkierung von  $g(t)$  ist dann

$g_P(p) = \tilde{p} : B_0 \rightarrow \epsilon$ ,

für das synthetische  
Attribut von  $B_0$  ist die Berechnungsvorschrift  $Term_N(ext(rank_\Sigma(B_0)), H)$  festgelegt.

Also  $s = Term_N(ext(rank_\Sigma(B_0)), H) = Term_N(H) = Term_N(H_t)\checkmark$ .

$h \rightarrow h + 1$ :

Sei bzw. seien

- $p = (B_0, H)$  die Markierung des Wurzelnotens  $k$  von  $t$ ,
- $ord_N(H) = e_1, \dots, e_n$  mit  $lab(e_i) = B_i$ ,
- $k_1, \dots, k_n$  ( $n = rank_\Sigma(B_0) - 1$ ) die Nachfolgerknoten von  $k$  (von links nach rechts) und
- $t_i$  ( $1 \leq i \leq n$ ) der Ableitungsteilbaum mit  $k_i$  als Wurzel.
- $h_0^0 = Term_N(H)$ .
- $\tilde{k}$  der Wurzelknoten von  $g(t)$ , dann ist
- $g_P(p) = \tilde{p} : B_0 \rightarrow B_1 \dots B_n$  die Markierung von  $\tilde{K}$ .
- $\tilde{k}_1, \dots, \tilde{k}_n$  die Nachfolgerknoten von  $\tilde{k}$  (von links nach rechts) und
- $\tilde{t}_i$  ( $1 \leq i \leq n$ ) der Ableitungsteilbaum mit  $\tilde{k}_i$  als Wurzel.

- Sei  $s_0^0$  die in  $\tilde{p}$  vereinbarte Berechnungsvorschrift für das synthetische Attribut von  $B_0$ .

Dann gilt:

$$h_0^0 = \text{Term}_N(H) = \text{Term}_N(\text{ext}(\text{rank}(H)), H) = \text{Term}_N(\text{ext}(\text{rank}_\Sigma(B_0)), H) = s_0^0.$$

Jetzt werden nacheinander die durch die Ableitungsteilbäume  $t_n, \dots, t_1$  (bzw.  $\tilde{t}_n, \dots, \tilde{t}_1 = g(t_n), \dots, g(t_1)$ ) festgelegten Terme eingefügt:

$t_n$  :

Sei  $\tilde{H}_n$  der durch  $t_n$  festgelegte Hypergraph (Dschungel mit  $\text{rank}_\Sigma(B_n) - 1 = |\text{Inh}(B_n)|$  Variablen).

Sei  $H_n$  der Dschungel, der entsteht, wenn in  $H$  die Hyperkante  $e_n$  durch  $\tilde{H}_n$  ersetzt wird, und sei  $h_n = \text{Term}_N(H_n)$ .

$h_n$  kann auch wie folgt bestimmt werden:

Sei  $\tilde{h}_n = \text{Term}_N(\tilde{H}_n)$ , dann

sei  $h_n^{|\text{Inh}(B_0)|}$  der Term, der entsteht, wenn in  $h_0^0 A_0^{B_n}$  durch  $\tilde{h}_n$  ersetzt wird.

Sei  $h_n^{i-1}$  ( $i \geq 1$ ) der Term, den man erhält, wenn man in  $h_n^i A_0^{B_i}$  durch  $\text{Term}_N(\text{nod}(e_n, i), H)$  ersetzt.

Dann ist  $h_n^0 = h_n$ .

$\tilde{t}_n$  :

Sei  $\tilde{s}_n^0$  der Term, der dem synthetischen Attribut von  $B_n$  durch  $\tilde{t}_n$  zugeordnet wird.

Seien  $\tilde{s}_n^i$ ,  $1 \leq i \leq |\text{Inh}(B_n)|$ , die den inheriten Attributen von  $B_n$  zugeordneten Terme (die in  $g_P(p) = \tilde{p}$  als Berechnungsvorschriften gegeben sind).

Dann sei  $s_n$  der Term, der durch Einsetzen der  $\tilde{s}_n^i$ ,  $0 \leq i \leq |\text{Inh}(B_n)|$ , wie folgt entsteht:

Sei  $s_n^{|\text{Inh}(B_n)|}$  der Term, den man erhält, wenn man in  $s_0^0$  (jedes)  $A_0^{B_n}$  durch  $\tilde{s}_n^0$  ersetzt.

Sei  $s_n^{i-1}$  ( $i \geq 1$ ) der Term, den man erhält, wenn man in  $s_n^i$  (jedes)  $A_i^{B_n}$  durch  $\tilde{s}_n^i$  ersetzt.

Dann sei  $s_n = s_n^0$ .

Es gilt  $s_n^i = h_n^i$  für  $0 \leq i \leq |\text{Inh}(B_0)|$  und

$$s_n = s_n^0 = h_n^0 = h_n.$$

In gleicher Art und Weise werden die Ableitungsteilbäume  $t_{n-1}, \dots, t_1$  bzw.  $\tilde{t}_{n-1}, \dots, \tilde{t}_1$  behandelt:

$t_i$  ( $1 \leq i \leq |\text{Inh}(B_0)| - 1 = n$ ) :

Sei  $\tilde{H}_i$  der durch  $t_i$  festgelegte Hypergraph.

Sei  $H_i$  der Dschungel, der entsteht, wenn man in  $H_{i+1}$  die Hyperkante  $e_i$  durch  $\tilde{H}_i$  ersetzt und sei  $h_i = \text{Term}_N(H_i)$ .

$\tilde{t}_i$  :

Sei  $\tilde{s}_i^0$  der Term, der dem synthetischen Attribut von  $B_i$  durch  $\tilde{t}_i$  zugeordnet wird.

Seien  $\tilde{s}_i^j$ ,  $1 \leq j \leq |\text{Inh}(B_i)|$ , die den inheriten Attributen von  $B_i$  zugeordneten

Terme (Berechnungsvorschriften in  $\tilde{p}$ ).

Sei  $s_i^{Inh(B_i)}$  der Term, den man erhält, wenn man in  $s_{i+1}$  jedes  $A_0^{B_i}$  durch  $\tilde{s}_i^0$  ersetzt.

Sei  $s_i^{j-1}$  ( $j \geq 1$ ) der Term, den man erhält, wenn man in  $s_i^j$  jedes  $A_j^{B_i}$  durch  $\tilde{s}_i^j$  ersetzt und

sei  $s_i = s_i^0$ .

Dann gilt  $h_i = s_i$ .

Insbesondere gilt

$$h_1 = s_1,$$

$$h_1 = Term(H_1) \text{ und}$$

$H_1$  ist der Hypergraph, den man erhält, wenn man alle durch die Ableitungsteilbäume  $t_n, \dots, t_1$  festgelegten Hypergraphen „eingesetzt“ hat, also ist  $H_1$  gerade der durch  $t$  festgelegte Hypergraph  $H_t$  (Dschungel mit  $|Inh(B_0)|$  Variablen).

$H_1$  enthält nur terminalen Hyperkanten.

Also gilt  $Term_N(H_t) = Term_N(H_1) = h_1 = s_1$ .

Bleibt zu betrachten, wo der Unterschied zwischen  $Term$  und  $Term_N$  liegt:

- Knoten mit nichtterminalen Hyperkanten als Eingangskanten werden unterschiedliche Terme zugewiesen.
- Die externen Knoten – bis auf dem höchsten externen Knoten – werden verschieden gehandhabt.

Ist  $t$  ein Ableitungsbaum, dann legt  $t$  allerdings einen terminalen Dschungel ohne Variablen fest.

In diesem Falle gilt also  $Term(H_t) = Term_N(H_t) = s_1$ .

□

**Satz 5.20** Sei  $G$  eine Hypergraph-Grammatik und  $\tilde{G}$  die dazu konstruierte L-attributierte Grammatik.

Dann gilt :  $T_{HG}(G) = T_{LaG}(\tilde{G})$ .

**Beweis:**

„ $\subseteq$ “:  $s \in T_{HG}(G)$

$\Rightarrow$  es existiert ein Ableitungsteilbaum  $t$  zu  $G$ , der einen Hypergraphen  $H$  festlegt, zu dem der Term  $s$  assoziiert wird.

$\Rightarrow$  (mit vorherigen Satz)

$f(t)$  ist Ableitungsbaum zu  $\tilde{G}$  und legt den Term  $s$  zur Berechnung des designierten Attributs fest.

„ $\supseteq$ “:  $s \in T_{LaG}(\tilde{G})$

$\Rightarrow$  es existiert ein Ableitungsteilbaum  $t$  zu  $\tilde{G}$ , der  $s$  zur Berechnung des designierten Attributs festlegt.

$\Rightarrow$  der markierte Baum  $f^{-1}(t)$  ist Ableitungsbaum zu  $G$ , mit dem durch  $f^{-1}(t)$  festgelegten Hypergraphen wird der Term  $s$  assoziiert.

Also gilt  $T_{HG}(G) \subseteq T_{LaG}(\tilde{G})$  und  $T_{HG}(G) \supseteq T_{LaG}(\tilde{G})$  und somit  $T_{HG}(G) = T_{LaG}(\tilde{G})$ .

□

Aus dem Satz 5.20 folgt dann unmittelbar:

**Folgerung 5.21**  $\mathcal{T}_{HG} \subseteq \mathcal{T}_{LaG}$

Zusammen mit Folgerung 5.14 also

**Folgerung 5.22**  $\mathcal{T}_{HG} = \mathcal{T}_{LaG}$



# Kapitel 6

## IO-Makrogrammatiken und kontextfreie Hypergraph-Grammatiken

### 6.1 Simulation von kontextfreien Hypergraph-Grammatiken mit Dschungel-Eigenschaft durch IO-Makrogrammatiken

Es soll gezeigt werden, daß die Menge der Termsprachen der Hypergraph-Grammatiken mit Dschungel-Eigenschaft Teilmenge der IO-Makrosprachen ist.

Um dies zu zeigen, wird zunächst eine Konstruktionsbeschreibung angegeben, die zu einer gegebenen Hypergraph-Grammatik eine termgenerierende IO-Makrogrammatik liefert. Dann wird gezeigt, daß die konstruierte Grammatik die gleiche Termsprache besitzt wie die vorgelegte Hypergraph-Grammatik.

Wie im Falle der L-attribuierten Grammatik wird bei der Konstruktion jeder nicht-terminalen Hyperkante eineindeutig ein Funktionsname zugeordnet.

Dabei findet sich die Zahl der Tentakel einer nichtterminalen Hyperkante  $B$  (also  $rank_{\Sigma}(B)$ ) in der Anzahl der Argumente der Funktion  $B$  ( $\rho(B)$ ) wieder – sie ist allerdings um eins vermindert, da ja das höchste Tentakel sozusagen das Ergebnistentakel ist ( $rank_{\Sigma}(B) - 1 = \rho(B)$ ).

Zu einer Hypergraph-Produktion werden allerdings möglicherweise mehrere IO-Makro-Produktionen konstruiert. Befinden sich  $n$  nichtterminale Hyperkanten auf der rechten Seite der Hypergraph-Produktion  $p$ , so werden dazu  $n + 1$  IO-Makro-Produktionen konstruiert werden.

Daß eine Bijektion zwischen den Produktionen wie im L-attribuierten Fall wohl nicht möglich ist, wird vielleicht schon an dem folgenden Beispiel deutlich:

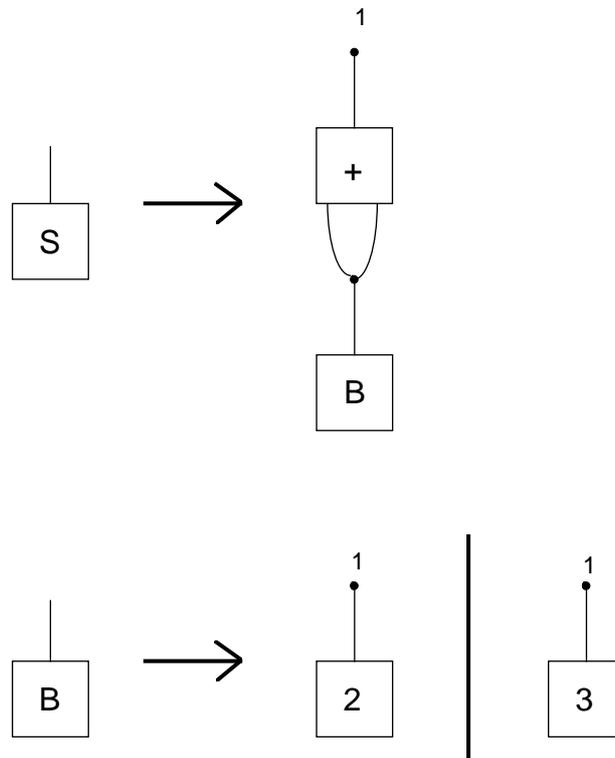


Abbildung 6.1: Hypergraph-Produktionen

**Beispiel 6.1** Sei  $G = (\Sigma, \Delta, P, S)$  eine Hypergraph-Grammatik mit den Produktionen aus Abbildung 6.1. Mit dieser Grammatik können nur zwei terminale Hypergraphen erzeugt werden und die beiden dadurch festgelegten Terme sind  $+(2, 2)$  und  $+(3, 3)$ .

Dazu werden die folgenden IO-Makro-Produktionen konstruiert werden:

- $S() = A_1^1(B)$
- $B() = 2$
- $B() = 3$
- $A_1^1(x_1) = +(x_1, x_1)$

### 6.1.1 Konstruktion

**Konstruktionsbeschreibung:**

Gegeben eine Hypergraph-Grammatik

$G = (\Sigma, \Delta, P, S)$  mit Dschungel-Eigenschaft,

$P = \{p^1, \dots, p^n\}$ ,  $p^i = (B^i, H^i)$  und

$ord_N(H^i) = e_1^i, \dots, e_{n_i}^i$  eine Aufzählung aller Hyperkanten aus  $H^i$  mit nichtterminalem Label, die die  $v$ -Ordnung einbettet.

Konstruiere IO-Makrogrammatik  $\tilde{G} = (\Sigma_M, F, V, \rho, S_M, \Pi)$  mit

- $\Sigma_M = \Delta$
- $F = F_1 \cup F_2$ 
  - $F_1 = \Sigma - \Delta$  (also zu jedem nichtterminalen Label existiert ein Funktionsname)
  - $F_2 = \bigcup_{i=1}^n \{A_1^i, \dots, A_{n_i}^i\}$  (die  $A_j^i$  seien „neue“ Zeichen:  $F_1 \cap F_2 = \emptyset$ )  
(für jedes dieser „Hilfszeichen“ wird später genau ein Makro angegeben werden)
- $V = \{x_1, \dots, x_k, y_1, \dots, y_l\}$ 
  - $k = \max\{n_i \mid 1 \leq i \leq n\}$
  - $l = \max\{\text{rank}_\Sigma(B) - 1 \mid B \in \Sigma - \Delta\}$
- Sei  $C$  aus  $F$ :
 
$$\rho(C) = \begin{cases} \text{rank}_\Sigma(C) - 1 & \text{falls } C \in F_1 \\ \text{rank}_\Sigma(C^i) - 1 + j & \text{falls } C = A_j^i \in F_2 \end{cases}$$
- $S_M = S$

Bleibt die Beschreibung der Produktionen.

Dazu definieren wir zunächst  $Term_M$  und  $Term_E$ .

$Term_M$  ( $M$  wie Makro) ist eine „Variante“ des assoziierten Terms – Hyperkanten mit nichtterminalen Label werden abweichend behandelt.

$Term_E$  ( $E$  wie Einstieg) dient als Einstieg in die rekursive  $Term_M$ -Formel:

(Gegeben  $\Sigma$  und  $\Delta$ , sei  $H$  ein Dschungel,  $ord_N(H) = e_1, \dots, e_n$  eine gegebene Aufzählung der Hyperkanten mit nichtterminalen Label, die die  $v$ -Ordnung einbettet, dann ist:)

**Definition 6.2** ( $Term_M$ ) Sei  $v$  ein Knoten aus  $H$ :

$$Term_M(v, H) = \begin{cases} y_i & \text{falls } v = ext(i), i \leq rank(H) - 1 \\ x_i & \text{falls } e_i \text{ Eingangskante von } v \\ lab(e) & \text{falls } e \text{ (terminale) Eingangskante von} \\ & \text{v und } rank_\Sigma(e) = 1 \\ f(t_1, \dots, t_m) & \text{falls } e \text{ (terminale) Eingangskante von} \\ & \text{v, } rank_\Sigma(e) > 1, lab(e) = f \text{ und} \\ & t_i = Term_M(nod(e, i), H), 1 \leq i \leq rank_\Sigma(f) - 1 = m \end{cases}$$

und

**Definition 6.3** ( $Term_E$ ) Sei  $v$  ein Knoten aus  $H$ :

$$Term_E(v, H) = \begin{cases} B_j & \text{falls } e_j (j \in [1 : n]) \text{ (nichtterminale) Eingangskante von} \\ & \text{v, } lab(e_j) = B_j \text{ und } rank_\Sigma(B_j) = 1 \\ B_j(t_1, \dots, t_m) & \text{falls } e_j (j \in [1 : n]) \text{ (nichtterminale) Eingangskante von} \\ & \text{v, } lab(e_j) = B_j, rank_\Sigma(B_j) > 1 \text{ und} \\ & t_i = Term_M(nod(e_j, i), H), 1 \leq i \leq rank_\Sigma(B_j) - 1 = m \\ undefiniert & \text{sonst} \end{cases}$$

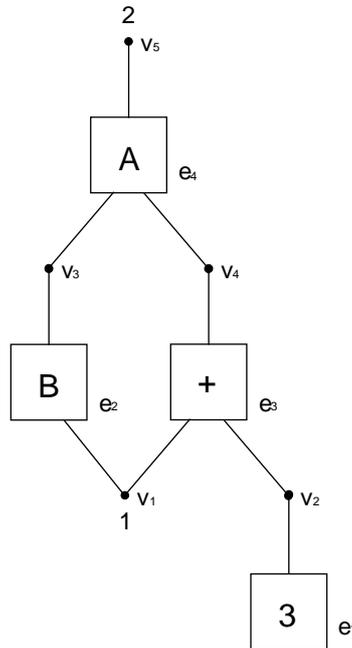
Zu diesen Definitionen geben wir ein kleines Beispiel:

**Beispiel 6.4** Sei  $H$  der Dschungel aus Abbildung 6.2.

Es muß  $ord_N(H) = e_2 e_4$  gelten, da  $e_2$  Vorgänger von  $e_4$  ist.  
Dann ist

- $Term_M(v_1, H) = y_1$
- $Term_M(v_2, H) = 3$
- $Term_M(v_3, H) = x_1$
- $Term_M(v_4, H) = +(Term_M(v_1, H), Term_M(v_2, H)) = +(y_1, 3)$
- $Term_M(v_5, H) = x_2$

und

Abbildung 6.2: Hypergraph und  $Term_M$  bzw.  $Term_E$ 

- $Term_E(v_1, H)$  undefiniert
- $Term_E(v_2, H)$  undefiniert
- $Term_E(v_3, H) = B(Term_M(v_1, H)) = B(y_1)$
- $Term_E(v_4, H)$  undefiniert
- $Term_E(v_5, H) = A(Term_M(v_3, H), Term_M(v_4, H)) = A(x_1, +(y_1, 3))$

Jetzt können wir die Produktionen beschreiben.

(Beachte:

$m_i$  entspricht dem Rang von  $B^i - 1$ ,

$n_i$  entspricht der Anzahl der nichtterminalen Symbole auf der rechten Seite von  $p^i$ )

- Sei  $p^i = (B^i, H^i)$ ,
- $ord_N(H^i) = e_1^i, \dots, e_{n_i}^i$ ,
- $lab(e_j^i) = B_j^i$ ,
- $rank_\Sigma(B^i) - 1 = \rho(B^i) = m_i$ .

Dann ist,

**falls**  $n_i = 0$  gilt, die Produktion

$$\bullet p_0^i : B^i(y_1, \dots, y_{m_i}) \rightarrow Term_M(ext(m_i + 1), H^i)$$

in  $\Pi$ .

**Sonst** ( $n_i \geq 1$ ) sind die Produktionen

$$\begin{aligned} \bullet p_0^i & : B^i(y_1, \dots, y_{m_i}) \rightarrow A_1^i(Term_E(nod(e_1^i, rank_\Sigma(B_1^i)), H^i), y_1, \dots, y_{m_i}) \\ \bullet p_j^i & : A_{j+1}^i(x_1, \dots, x_j, Term_E(nod(e_{j+1}^i, rank_\Sigma(B_{j+1}^i)), H^i), y_1, \dots, y_{m_i}) \text{ f\"ur } 1 \leq j \leq \\ & n_i - 1 \\ \bullet p_{n_i}^i & : A_{n_i}^i(x_1, \dots, x_{n_i}, y_1, \dots, y_{m_i}) \rightarrow Term_M(ext(m_i + 1), H^i) \end{aligned}$$

in  $\Pi$ .

$\Pi$  besteht aus genau den so konstruierten Produktionen:

$$\Pi = \bigcup_{i=1}^n \{p_0^i, \dots, p_{n_i}^i\}$$

Damit ist die Konstruktionsbeschreibung abgeschlossen.

### 6.1.2 Zwischenüberlegungen

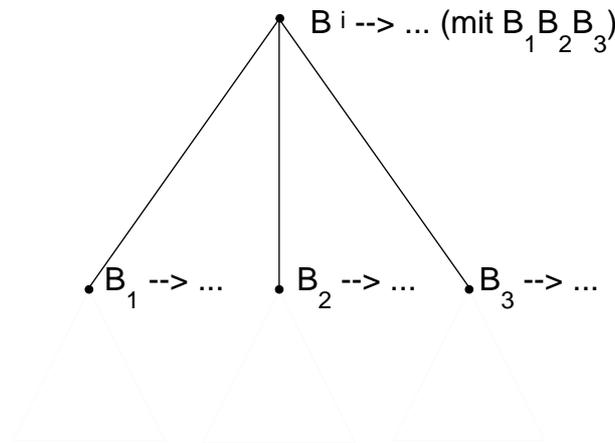
$Term_M$  liefert Terme, in denen keine nichtterminalen Symbole auftreten. Folglich liefert  $Term_E$  Terme, in denen genau ein nichtterminales Symbol auftritt.

Die konstruierten Produktionen sind also entweder terminal, oder auf der rechten Seite treten genau zwei nichtterminale Symbole auf (in der Produktion  $p_j^i$ ,  $1 \leq i \leq n$ ,  $0 \leq j \leq n_i - 1$ , sind dies  $A_{j+1}^i$  und  $B_{j+1}^i$ ).

Außerdem gilt, daß das Symbol  $A_{j+1}^i$  überhaupt nur in zwei Produktionen auftaucht, nämlich auf der rechten Seite von  $p_j^i$  und auf der linken von  $p_{j+1}^i$ . Und in der Produktion  $p_j^i$  befindet sich  $B_{j+1}^i$  „im Inneren“ von  $A_{j+1}^i$ , d. h. es wird immer (IO-Makro)  $B_{j+1}^i$  vor  $A_{j+1}^i$  ausgewertet.

Betrachten wir Ableitungsteilbäume für  $\tilde{G}$ , so können wir folgendes feststellen:

- Ein Knoten ist entweder Blatt, oder er hat sowohl einen linken als auch einen rechten Nachfolger (es handelt sich also um einen [speziellen] binären Baum).

Abbildung 6.3: Ausschnitt aus Ableitungsteilbaum mit  $p^i$  als Wurzel

- Ein Knoten ist genau dann ein Blatt, wenn er mit einer Produktion  $p_{n_i}^i$  ( $1 \leq i \leq n$ ) markiert ist.
- Ist ein Knoten mit  $p_j^i$  markiert und kein Blatt ( $\Rightarrow j < n_i$ ), dann ist der rechte Nachfolger mit  $p_{j+1}^i$  markiert – da dies die einzige Produktion mit  $A_{j+1}^i$  auf der linken Seite ist.  
Der linke Nachfolger ist mit einer Produktion  $p_0^k$  markiert (mit  $lhs(p_0^k) = B_{j+1}^i$ ) – hier gibt es eventuell mehrere Möglichkeiten.

Insbesondere können wir also für einen Knoten anhand seiner Markierung bestimmen, ob

- er einen rechten Nachfolger hat und
- falls er einen hat, welche Markierung dieser besitzt.

Offensichtlich haben rechte Nachfolger auch immer den gleichen „oberen“ Index wie ihre Vorgänger und einen um eins erhöhten „unteren“ Index. Betrachten wir dazu

**Beispiel 6.5** Sei  $p^i = (B^i, H^i)$ ,  $ord_N(H^i) = e_1 e_2 e_3$ ,  $lab(e_1) = B_1$ ,  $lab(e_2) = B_2$  und  $lab(e_3) = B_3$ , dann sieht ein Ableitungsteilbaum mit  $p^i$  als Wurzelmarkierung so wie in Abbildung 6.3 skizziert aus. Zu  $p^i$  werden die vier Produktionen  $\tilde{p}_0^i$ ,  $\tilde{p}_1^i$ ,  $\tilde{p}_2^i$  und  $\tilde{p}_3^i$  konstruiert und in Abbildung 6.4 ist der Wurzelbereich eines Ableitungsteilbaums mit  $\tilde{p}_0^i$  als Wurzelmarkierung skizziert.

Bezeichnen wir die Knotenfolge  $v_1, \dots, v_k$  in einem Ableitungsbaum zu  $\tilde{G}$  als „vollständige Rechtsfolge“ genau dann, wenn

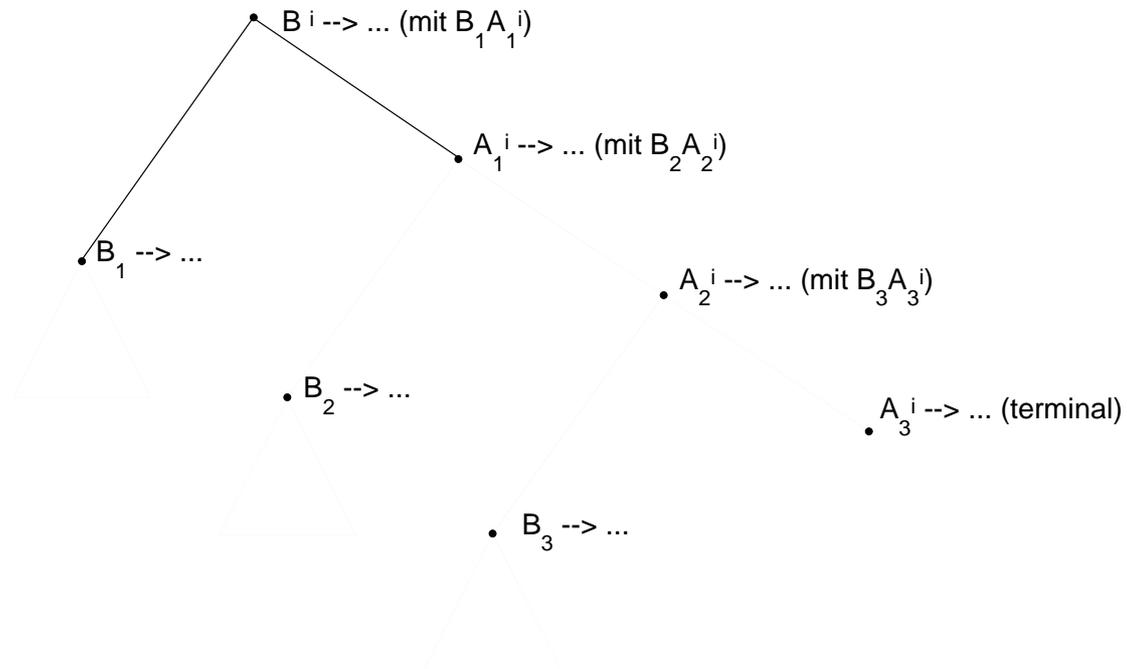


Abbildung 6.4: Ausschnitt aus Ableitungsteilbaum mit  $\tilde{p}_0^i$  als Wurzel

- $v_1$  linker Nachfolger seines Vorgängers,
- $v_{j+1}$  rechter Nachfolger von  $v_j$  ist ( $1 \leq j \leq k-1$ ) und
- $v_k$  ein Blatt ist,

dann gilt also: Es existiert ein  $i$ , so daß  $p_{j-1}^i$  die Markierung von  $v_j$  ist ( $1 \leq j \leq k$ ).  
Außerdem folgt:  $k = n_i - 1$ .

Wir können also ohne Informationsverlust diese Ableitungsbäume von  $\tilde{G}$  (bzw. auch Ableitungsteilbäume mit der Einschränkung, daß die Wurzelmarkierung als unteren Index eine 0 aufweist) „schrumpfen“ – um sie besser mit denen von  $G$  in Relation setzen zu können.

Wir beenden die Zwischenüberlegungen und definieren dieses Schrumpfen.

### 6.1.3 Weitere Vorbereitungen

**Definition 6.6 (shrink)** Ist  $t$  Ableitungsteilbaum zu  $\tilde{G}$  mit  $p_0^i$  als Wurzelmarkierung ( $1 \leq i \leq n$ ).

Sei  $V$  die Menge der Knoten von  $t$ ,

$E_l, E_r \subset V \times V$  die Menge der Kanten, die einen Knoten mit seinem linken resp. rechten Nachfolger verbinden und

$lab$  die Markierungsfunktion von  $t$  ( $lab : V \rightarrow \Pi$ ).

Dann ist  $shrink(t)$  der markierte Baum mit

- $V_{shrink} = \{v \in V \mid \exists j \in \mathbb{N} : lab(v) = p_0^j\}$  (also alle Knoten, die nicht rechter Nachfolger eines anderen sind),
- $E_{shrink} = \{(v, v') \in V_{shrink} \times V_{shrink} \mid \exists v_1, \dots, v_m \in V, k \in \mathbb{N} : v = v_1 \wedge v' = v_m \wedge (\forall j \in [1 : m - 2] : (v_j, v_{j+1}) \in E_r) \wedge (v_{m-1}, v_m) \in E_l\}$  ( $v$  ist der nächste „noch existierende“ Vorgänger von  $v'$ ) und
- $lab_{shrink}(v) = p^i \Leftrightarrow lab(v) = p_0^i$ .

Nicht ohne Grund liefert  $lab_{shrink}$  Marken, die einen Ableitungsteilbaum zu  $G$  markieren können:

**Satz 6.7** Sei  $t$  ein Ableitungsteilbaum zu  $\tilde{G}$  mit  $p_0^i$  als Wurzelmarkierung  $\Rightarrow shrink(t)$  ist Ableitungsteilbaum zu  $G$ .

**Beweis:**

Induktion über die Höhe  $h$  von  $shrink(t)$ .

$h = 1$ :

$shrink(t)$  besteht nur aus einem Knoten  $k$

$\Rightarrow t$  besteht nur aus einem Knoten, also ist  $p_0^i$  eine terminale Produktion

$\Rightarrow p^i$  ist eine terminale Produktion (und Markierung von  $k$ )

$\Rightarrow shrink(t)$  ist Ableitungsteilbaum zu  $G$ .

$h \rightarrow h + 1$ :

**Überlegung:** Wir haben die Markierung der Wurzel von  $t$ . Daraus können wir rekursiv folgern,

- ob der Knoten einen rechten Nachfolger hat und
- wie dieser gegebenenfalls markiert ist.

Für jeden dieser rechten Nachfolger – und für die Wurzel selbst – können wir folgern,

- ob dieser Knoten einen linken Nachfolger hat (nur der letzte Knoten der „Rechtsfolge“ besitzt keinen) und

- wie die linke Seite der den linken Nachfolger markierenden Produktion aussieht.

Für  $shrink(t)$  können wir direkt die Wurzelmarkierung aus der Wurzelmarkierung von  $t$  folgern. Aus den obigen Folgerungen können wir ferner schließen,

- wieviele Nachfolger die Wurzel von  $shrink(t)$  hat und
- wie die linke Seite einer Markierung eines Nachfolgers der Wurzel aussieht.

Damit können wir prüfen, ob die Wurzel von  $shrink(t)$  die Bedingungen erfüllt, um Knoten eines Ableitungsteilbaums zu sein.

Für die übrigen kann dies per Induktionsannahme geschlossen werden.

Sei  $t$  ein Ableitungsteilbaum zu  $\tilde{G}$  mit der Wurzelmarkierung  $p_0^i$  und der Eigenschaft, daß  $shrink(t)$  die Höhe  $h + 1$  hat.

Sei  $k$  die Wurzel von  $shrink(t)$ , dann ist  $k$  mit  $p^i = (B^i, H^i)$  markiert.

Sei  $ord_N(H_i) = e_1^i \dots e_{n_i}^i$  und  $\tilde{k}_0$  die Wurzel von  $t$ , dann folgt aus der Konstruktion, daß  $\tilde{k}_0$   $n_i$  „rechte“ Nachfolger hat, d. h. in  $t$  existieren die Knoten  $\tilde{k}_j^r$  ( $1 \leq j \leq n_i$ ) mit  $\tilde{k}_1^r$  ist rechter Nachfolger von  $\tilde{k}_0$  und  $\tilde{k}_j^r$  ist rechter Nachfolger von  $\tilde{k}_{j-1}^r$ . Außerdem ist klar, daß  $\tilde{k}_j^r$  mit  $p_j^i$  markiert ist und  $\tilde{k}_{n_i}^r$  ein Blatt ist.

(\*) Also hat  $k$  (Wurzel von  $shrink(t)$ )  $n_i$  Nachfolger,

die wir mit  $k_1, \dots, k_{n_i}$  bezeichnen.

Sei  $\tilde{k}_j^l$  der linke Nachfolger von  $\tilde{k}_{j-1}^r$  ( $\tilde{k}_{j-1}^r$  ist mit  $p_{j-1}^i : A_{j-1}^i(\dots) = A_j^i(\dots, B_j^i(\dots), \dots)$  markiert) und  $\tilde{p}_j$  seine Markierung, dann ist  $lhs(\tilde{p}_j) = B_j^i$ .

Sei  $p_j$  die Markierung von  $k_j$ , dann folgt damit:

$$(**) lhs(p_j) = B_j^i$$

Also erfüllt die Wurzel  $k$  von  $shrink(t)$  die Bedingungen, Knoten eines Ableitungsteilbaums zu  $G$  zu sein (richtige Anzahl von Nachfolgern (\*)) und die Nachfolger sind geeignet markiert (\*\*).

Für die übrigen Knoten kann dies per Induktionsannahme geschlossen werden.

□

**Satz 6.8** Sei  $t$  Ableitungsteilbaum zu  $G$ .

Dann existiert ein markierter Baum  $\tilde{t}$  mit

- $shrink(\tilde{t}) = t$  und
- $\tilde{t}$  ist Ableitungsteilbaum zu  $\tilde{G}$ .

**Beweis:**

Induktion über die Höhe  $h$  von  $t$ .

$h = 1$ :

Terminal = trivial.

$h \rightarrow h + 1$ :

Sei  $k$  der Wurzelknoten von  $t$  mit der Markierung  $p^i$  und den Nachfolgerknoten  $k_1, \dots, k_{n_i}$ .

Sei  $t_j$  der Ableitungsteilbaum mit  $k_j$  als Wurzel (Teil von  $t$ ).

Dann („*unshrink*“):

Sei  $k_0^r$  die Wurzel von  $\tilde{t}$ , rechts folgen  $k_1^r, \dots, k_{n_i}^r$ .

Linker Nachfolger von  $k_{j-1}^r$  sei  $k_j^l$  ( $1 \leq j \leq n_i$ ). An  $k_j^l$  hänge  $\tilde{t}_j$  (mit  $shrink(\tilde{t}_j) = t_j$ , per Induktionsannahme existent). Die Markierung von  $k_j^r$  sei  $p_j^i$ .

Dann gilt

- $shrink(\tilde{t}) = t$  und
- $\tilde{t}$  ist Ableitungsteilbaum zu  $\tilde{G}$ .

□

### 6.1.4 Nachweis der Gleichheit

**Satz 6.9** Sei  $t$  ein Ableitungsbaum zu  $\tilde{G}$  und sei  $s$  der durch  $t$  festgelegte Term.

Sei  $H$  der durch  $shrink(t)$  – Ableitungsbaum zu  $G$  – festgelegte Dschungel.

Dann gilt :  $Term(H) = s$ .

**Beweis:**

Wir zeigen:

Sei  $t$  ein Ableitungsteilbaum zu  $\tilde{G}$

mit der Wurzelmarkierung  $p_0^i$  und

sei  $s$  der durch  $t$  festgelegte Term.

Sei  $H$  der durch  $shrink(t)$  – Ableitungsteilbaum zu  $G$  – festgelegte Dschungel (mit  $rank_\Sigma(B_i) - 1$  Variablen).

Dann gilt :  $Term(H) = s$ .

Induktion über die Höhe  $h$  von  $shrink(t)$ .

$h = 1$ :

$p^i = (B^i, H^i)$  ist eine terminale Produktion, also ist auch  $p_0^i$  eine terminale Produktion, nämlich  $B^i(y_1, \dots, y_{m_i}) = Term_M(ext(m_i + 1), H^i)$ .

Da in  $H^i$  nur Hyperkanten mit terminalem Label auftreten, gilt „ $Term_M = Term$ “, also

$$\begin{aligned}
 s &= B^i(y_1, \dots, y_{m_i}) \\
 &= Term_M(ext(m_i + 1), H^i) \\
 &= Term(ext(m_i + 1), H^i) \\
 &= Term(ext(rank_\Sigma(B^i)), H^i) \\
 &= Term(H^i) \\
 &= Term(H)
 \end{aligned}$$

$h \rightarrow h + 1$ :

Sei

- $k$  der mit  $p^i = (B^i, H^i)$ ,  $ord_N(H^i) = e_1^i \dots e_{n_i}^i$ , markierte Wurzelknoten von  $shrink(t)$  und seien  $k_1, \dots, k_{n_i}$  die Nachfolgerknoten von  $k$ .
- $t_j$  der Teil von  $shrink(t)$ , der an  $k_j$  hängt, also jeweils wieder ein Ableitungsteilbaum zu  $G$  (mit einer Höhe  $\leq h$ ).
- $H_j$  der durch  $shrink(t)$  festgelegte Dschungel mit  $rank_\Sigma(B_j^i) - 1$  Variablen.
- $H$  der durch  $shrink(t)$  festgelegte Dschungel mit  $rank_\Sigma(B^i) - 1$  Variablen.

Setze:

1.  $h_j^0 = Term(H_j)$  für  $j \in [1 : n_i]$
2.  $h_j^1 = „h_j^0$ , wobei die  $y_k$  in  $h_j^0$  durch  $y_k^j$  ersetzt werden“  $k \in [1 : rank(e_j^i) - 1]$  (diese Ersetzung wird gemacht, damit man bspw. das  $y_1$  von  $Term(H_3)$  vom  $y_1$  von  $Term(H_4)$  unterscheiden kann – vorbereitend für den nächsten Schritt:)
3.  $h_j^2 = „h_j^1$ , wobei  $y_k^j$  ersetzt wird durch:  $Term_M(nod(e_j^i, k), H^i)“$  ( $j$  und  $k$  wie oben).

$H$  kann offensichtlich wie folgt konstruiert werden:

1.  $\tilde{H}_{n_i} = H^i$ .
2.  $\tilde{H}_{j-1}$  ergibt sich aus  $\tilde{H}_j$ , indem dort die Hyperkante  $e_j^i$  durch  $H_j$  ersetzt wird ( $1 \leq j \leq n_i$ ).
3. Dann ist  $H = \tilde{H}_0$ .

Entsprechend kann  $Term(H)$  konstruiert werden:

1.  $h_{n_i}^3 = Term_M(H^i)$ .
2.  $h_{j-1}^3 = „h_j^3$ , wobei  $x_j$  durch  $h_j^2$  ersetzt wird“.
3. Dann ist  $Term(H) = h_0^3$ .

Sei

- $\tilde{k}_0^i$  der (mit  $p_0^i$  markierte) Wurzelknoten von  $t$ ,

- $\tilde{k}_j^r$  der (mit  $p_j^i$  markierte) rechte Nachfolger von  $k_{j-1}^{\tilde{r}}$  ( $1 \leq j \leq n_i$ ) und
- $\tilde{k}_j^l$  der linke Nachfolger von  $k_{j-1}^{\tilde{r}}$ .
- $\tilde{t}_j$  der Teil von  $t$ , der an  $\tilde{k}_j^l$  hängt (– es gilt:  $shrink(\tilde{t}_j) = t_j$ ).

Betrachten wir nun die Produktionen  $p_0^i, \dots, p_{n_i}^i$  und die Variable  $x_k$  ( $1 \leq k \leq n_i$ ), so stellen wir fest, daß  $x_k$  das erste Mal in  $p_{k-1}^i$  auftritt, dort wir sie „gesetzt“, nämlich auf  $Term_E(nod(e_k^i, rank(e_k^i)), H^i)$ . Danach wird sie „weitergereicht“ ( $x_k$  wird auf  $x_k$  gesetzt):

$$p_{k-1}^i : A_{k-1}^i(x_1, \dots, x_{k-1}, y_1, \dots, y_{m_i}) = A_k^i(x_1, \dots, x_{k-1}, Term_E(s.o.), y_1, \dots, y_{m_i})$$

$$p_j^i : A_j^i(x_1, \dots, x_k, \dots) = A_{j+1}^i(x_1, \dots, x_k, \dots) \text{ für } j \geq k$$

Untersuchen wir den „Wert“ von  $x_k$ :

In  $p_{k-1}^i$  wird

$$x_k = Term_E(nod(e_k^i, rank(e_k^i)), H^i)$$

$$= B_k^i(Term_M(nod(e_k^i, 1), H^i), \dots, Term_M(nod(e_k^i, rank(e_k^i) - 1), H^i))$$

gesetzt.

Der Ableitungsteilbaum  $\tilde{t}_k$  legt einen Term  $s_k^0$  mit  $\rho(B_k^i)$  Variablen ( $y_1, \dots, y_{\rho(B_k^i)}$ ) fest. Es sei

1.  $s_k^1 = s_k^0$ , wobei die  $y_l$  in  $s_k^0$  durch  $y_l^k$  ersetzt werden“.
2.  $s_k^2 = s_k^1$ , wobei  $y_l^k$  durch  $Term_M(nod(e_k^i, l), H^i)$  ersetzt wird“.

$x_k$  hat „nach Auswertung“ von  $\tilde{t}_j$  den Wert  $s_k^2$ .

Gleichzeitig gilt:

$$h_k^2 = s_k^2 \quad (\text{da } h_k^0 = s_k^0 \text{ nach Induktionsvoraussetzung})$$

Den durch  $t$  festgelegten Term  $s$  (mit  $\rho(B^i)$  Variablen) können wir dadurch herleiten, daß wir, beginnend bei  $Term_M(ext(m_i + 1), H^i)$ , nach und nach die  $x_k$  durch ihre „Werte“ ersetzen.

Setze:

- $s_{n_i}^3 = Term_M(ext(m_i + 1), H^i) = Term_M(H^i)$ .
- $s_{k-1}^3 = s_k^3$ , wobei  $x_k$  durch  $s_k^2$  ersetzt wird“ ( $1 \leq k \leq n_i$ ).
- $s = s_0^3$ .

Nun gilt aber  $s_k^3 = h_k^3$  und damit insbesondere

$$s = s_0^3 = h_0^3 = Term(H)$$

□

**Satz 6.10** Sei  $G$  eine Hypergraph-Grammatik und  $\tilde{G}$  die dazu konstruierte IO-Makrogrammatik.

Dann gilt :  $T_{HG}(G) = T_{IOM}(\tilde{G})$ .

**Beweis:**

„ $\subseteq$ “:  $s \in T_{HG}(G)$

$\Rightarrow$  es existiert ein Ableitungsteilbaum  $t$  zu  $G$ , der einen Hypergraphen  $H$  festlegt, zu dem der Term  $s$  assoziiert wird.

$\Rightarrow$  (Satz 6.8)

es existiert in Ableitungsteilbaum  $\tilde{t}$  zu  $\tilde{G}$  mit  $shrink(\tilde{t}) = t$ .

$\Rightarrow$  (Satz 6.9)

$\tilde{t}$  legt Term  $s$  fest (also  $s \in T_{IOM}(\tilde{G})$ ).

„ $\supseteq$ “:  $s \in T_{IOM}(\tilde{G})$

$\Rightarrow$  es existiert ein Ableitungsteilbaum  $\tilde{t}$  zu  $\tilde{G}$ , der  $s$  festlegt.

$\Rightarrow$  (Satz 6.7)

$shrink(\tilde{t})$  ist Ableitungsteilbaum zu  $G$ .

$\Rightarrow$  (Satz 6.9)

Sei  $H$  der durch  $shrink(\tilde{t})$  festgelegte Hypergraph, dann gilt  $Term(H) = s$ .

□

Aus dem Satz 6.10 folgt dann unmittelbar:

**Folgerung 6.11**  $T_{HG} \subseteq T_{IOM}$

## 6.2 Simulation von IO-Makrogrammatiken durch kontextfreie Hypergraph-Grammatiken mit Dschungel-Eigenschaft

In Umkehrung des vorherigen Abschnittes geben wir hier an, wie zu einer termgenerierenden IO-Makrogrammatik eine Hypergraph-Grammatik konstruiert werden kann, die die gleiche Termsprache besitzt.

Es wird darauf verzichtet, die Gleichheit der Termsprachen ausführlich zu beweisen – dies würde in Struktur (und Länge) den Beweisen aus den vorhergehenden Abschnitten sehr ähneln und damit den Leser langweilen.

### 6.2.1 Konstruktion

Gegeben eine termgenerierende IO-Makrogrammatik  $G_M = (\Sigma_M, \mathcal{F}, \mathcal{V}, \rho, S_M, \Pi)$ .

Konstruiere kontextfreie Hypergraph-Grammatik  $G = (\Sigma, \Delta, P, S)$  mit

- $\Sigma = \Sigma_M \cup \mathcal{F}$
- $\Delta = \Sigma_M$
- $P = \{p \mid \tilde{p} \in \Pi\}$ 
  - ist  $\tilde{p} : F(x_1, \dots, x_{\rho(F)}) \rightarrow t$  in  $\Pi$
  - dann ist  $p = (F, jung(t, \{x_1, \dots, x_{\rho(F)}\}))$  in  $P$
- $S = S_M$
- $rank_{\Sigma}(X) = \rho(X) + 1$

Ferner wird festgelegt:

Ist  $F_1, \dots, F_n$  ( $n \geq 0$ ) die Aufzählung der nichtterminalen Symbole in  $t$ , die für Ableitungsteilbäume verwendet wird (typischerweise von innen nach außen und innerhalb nebeneinanderstehender Symbole von links nach rechts), und ist  $e_i$  die zu  $F_i$  konstruierte Hyperkante, dann ist  $ord_N(jung(t, \dots)) = e_1 \dots e_n$  die durch die Konstruktion festgelegte Ordnung der nichtterminalen Symbole des konstruierten Hypergraphen.

### 6.2.2 Beweisidee

Eine Überprüfung der geforderten Eigenschaften zeigt, daß die konstruierte Hypergraph-Grammatik kontextfrei ist und die Dschungel-Eigenschaft besitzt, also eine kontextfreie Hypergraph-Grammatik mit Dschungel-Eigenschaft ist.

Wie in den vorangegangenen Konstruktionen resultiert hier nicht nur eine Bijektion zwischen den Produktionen, sondern auch eine zwischen den nichtterminalen Symbolen der linken Seiten der Produktionen aus den angegebenen Verfahren.

Aus diesen Bijektionen folgt eine Bijektion zwischen Ableitungsteilbäumen und über Induktion kann dann gezeigt werden, daß bei „gleichem“ Ableitungsbaum der gleiche Term festgelegt wird. Daraus kann dann wiederum die folgende Folgerung gezogen werden:

**Folgerung 6.12**  $\mathcal{T}_{IOM} \subseteq \mathcal{T}_{HG}$

Und mit Folgerung 6.11 ergibt sich dann

**Folgerung 6.13**  $\mathcal{T}_{HG} = \mathcal{T}_{IOM}$



# Kapitel 7

## Weitere Betrachtungen

### 7.1 Der Iterationssatz und IO-Makrogrammatiken

Für IO-Makrogrammatiken gibt es keinen Iterationssatz.

Für Hypergraph-Grammatiken allerdings gibt es einen und wir haben in Kapitel 6 ein Verfahren angegeben, um zu einer (termgenerierenden) IO-Makrogrammatik eine Hypergraph-Grammatik zu konstruieren.

Daher ist es naheliegend, nach sich hieraus ergebenden Zusammenhängen zu suchen. Die Betrachtungen, die daraus resultierten, werden im folgenden vorgestellt.

Schreibweisen:

Sei  $G = (\Sigma, \Delta, P, S)$  eine Hypergraph-Grammatik.

Dann sei  $HTerm(x_1, \dots, x_n)$  ein Term über  $\Delta \cup \{x_1, \dots, x_n\}$ . Soll oder kann die „Variablenmenge“ nicht näher angegeben werden, schreiben wir kurz auch  $HTerm$ .

Seien die in Abbildung 7.1 skizzierten Hypergraphen die Zerlegung eines Hypergraphen in *first*, *link* und *last* nach der Idee des Iterationssatzes (der  $n$ -te link-Knoten sei der aufwärts-link-Knoten). Die Bezeichnung der link-Knoten ist absichtlich so gewählt, daß  $x_i$  wiederholt verwendet wird. Welches  $x_i$  im Einzelfall gemeint ist, ergibt sich jeweils aus dem Zusammenhang. Jeder dieser Teilhypergraphen legt für diejenigen link-Knoten, für die er eine Eingangskante hat, einen  $HTerm$  (über eine Variablenmenge) fest – außerdem legt *first* einen  $Hterm(x_n)$  für den externen Knoten fest.

Wir können die (in der Abbildung skizzierten) Rechenregeln gewinnen:

- *first* legt  $HTerm$ e für  $x_1, \dots, x_{n-1}$  und einen  $HTerm(x_n)$  für den externen Knoten fest.

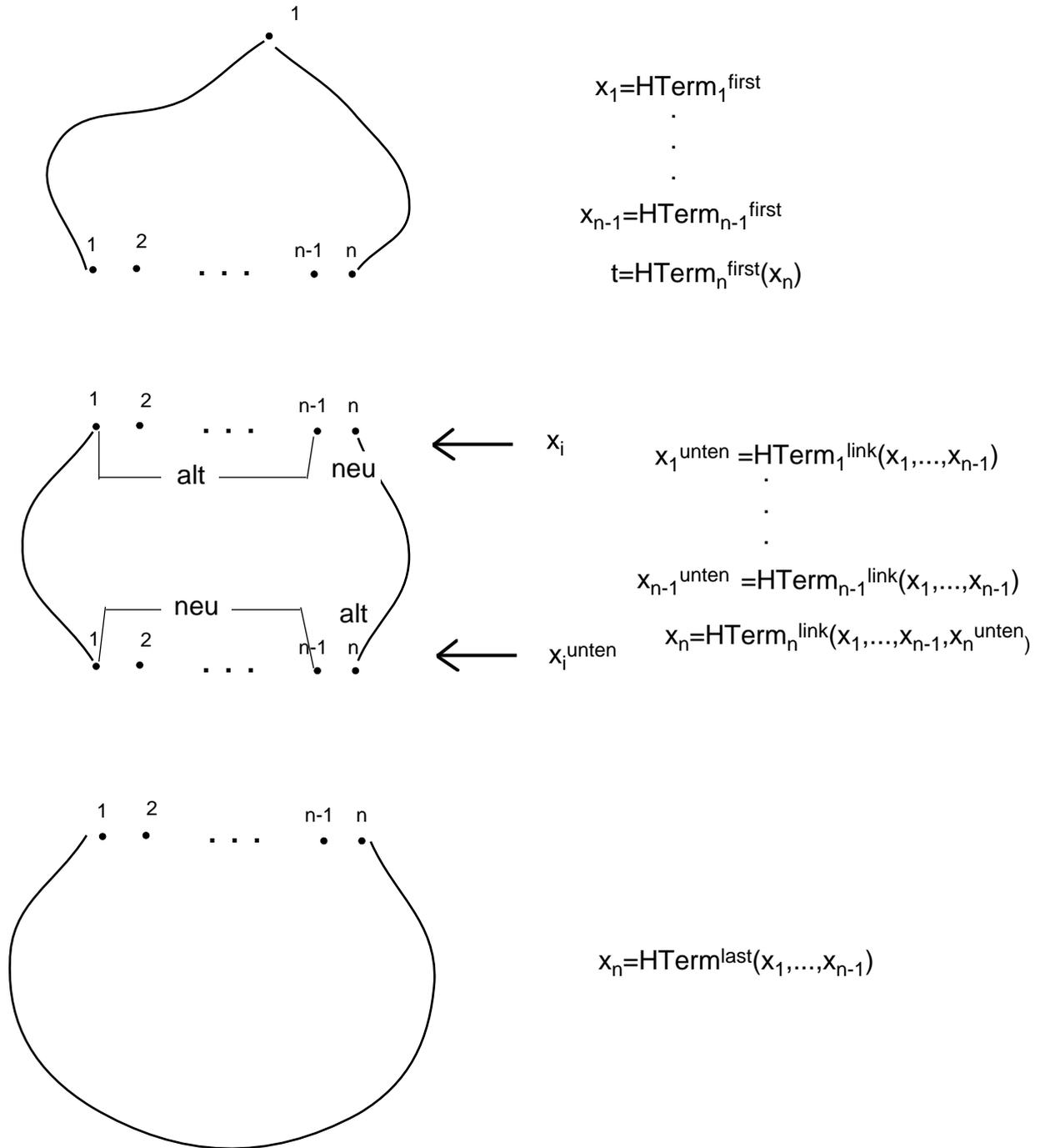


Abbildung 7.1: „Allgemeine“ Zerlegung in *first*, *link* und *last*

- *link* legt für die  $n - 1$  ersten unteren link-Knoten (für die abwärts-link-Knoten) *HTerm* fest, die nur von den oberen  $n - 1$  link-Knoten abhängen. Da der „Informationsfluß“ – wie schon die Namensgebung suggeriert – bei diesen link-Knoten abwärtsgerichtet ist, wurden in der Abbildung 7.1 die oberen mit „alt“ und die unteren mit „neu“ markiert.  
Also für  $x_i^{unten}$ ,  $1 \leq i \leq n - 1$ , wird in *link* ein  $HTerm(x_1, \dots, x_{n-1})$  festgelegt. In der „anderen“ Richtung wird für  $x_n$  ein  $HTerm(x_1, \dots, x_{n-1}, x_n^{unten})$  festgelegt – daß nur hier ein  $x_n$  bzw.  $x_n^{unten}$  auf der rechten Seite auftaucht ist Folge der Zyklenfreiheit.
- *last* legt für  $x_n$  einen  $HTerm(x_1, \dots, x_{n-1})$  fest.

Wichtig ist festzustellen, daß mittels dieser Rechenregeln der  $Term(first \otimes link \otimes last)$  in „getrennter Analyse“ der drei Hypergraphen *first*, *link* und *last* bestimmt werden kann:

### Startregeln

- Setze  $x_i := HTerm_i^{first}()$  für alle  $1 \leq i \leq n - 1$ .
- Setze  $t := HTerm_n(x_n)$ .

### Iterationsregeln

- Setze  $x_i^{unten} := HTerm_i^{link}(x_1, \dots, x_{n-1})$  für alle  $1 \leq i \leq n - 1$ .
- Werte  $x_i^{unten}$  zu  $HTerm()$  (ohne Variablen) aus ( $1 \leq i \leq n - 1$ ).
- Setze  $x_n := HTerm_n^{link}(x_1, \dots, x_{n-1}, x_n^{unten})$ .
- Werte  $x_n$  zu  $HTerm(x_n^{unten})$  aus.
- Werte  $t$  aus (ersetze jedes  $x_n$  durch den obigen Term).
- (Vorbereitungen für den nächsten Schritt:)
- Ersetze in  $t$  jedes  $x_n^{unten}$  durch  $x_n$ .
- Setze  $x_i := x_i^{unten}$  für  $1 \leq i \leq n - 1$ .

### Schlußregeln

- Setze  $x_n := HTerm^{last}(x_1, \dots, x_{n-1})$ .
- Werte  $x_n$  zu  $HTerm()$  aus.
- Werte  $t$  aus (zu  $HTerm$  ohne Variablen)

Dann ist  $Term(first \otimes link \otimes last) = t$ .

Bleibt die Antwort auf die Frage, warum wir uns diese Mühe gemacht haben bzw. was  $Term(first \otimes link^i \otimes last)$  ergibt:

Wenn wir einmal die Startregeln,  $i$ -mal die Iterationsregeln und schließlich einmal die Schlußregeln anwenden, ist  $t$  gerade  $Term(first \otimes link^i \otimes last)$  – jedenfalls behaupten wir dies und werden es noch an zwei Beispielen vorführen.

Für einen hinreichend großen Term  $t$ , der mittels einer termgenerierenden IO-Makrogrammatik  $G_M$  produziert werden kann, bedeutet dies, daß sich dann die obigen Regeln finden lassen, so daß einmaliges Anwenden jeder Regel gerade  $t$  liefert und auch bei mehrfacher Anwendung der Iterationsregel wieder einen Term aus  $L(G_M)$  entsteht.

Die Anzahl der einzelnen Regeln hängt lediglich vom Rang des nichtterminalen Symbols, „über“ das iteriert wird, ab.

**Beispiel 7.1** Sei  $G_M = (\{a\}, \{S, A\}, \{x\}, \rho, S, \Pi)$  eine IO-Makrogrammatik mit den Produktionen  $S = A(a)$ ,  $A(x) = A(xx)$  und  $A(x) = x$ . Es ist  $L(G_M) = \{a^{2^n} \mid n \geq 0\}$ .

Hierzu läßt sich eine termgenerierende IO-Makrogrammatik  $G_{term}$  bestimmen, die gerade solche Terme erzeugt, deren Auswertung Worte aus  $L(G_M)$  liefert.

Zu  $G_{term}$  wiederum kann eine Hypergraph-Grammatik  $G$  nach dem in Abschnitt 6.1 angegebenen Verfahren konstruiert werden. Deren Produktionen sehen wie in Abbildung 7.2 aus.

Nach Beseitigung der  $\epsilon$ -Produktion erhalten wir die Produktionen aus Abbildung 7.3 und wir gewinnen die Regeln:

**Startregeln**  $x_1 = a$ ,  $t = x_2$

**Iterationsregeln**  $x_1^{unten} = \circ(x_1, x_1) = x_1x_1$ ,  $x_2 = x_2^{unten}$

**Schlußregeln**  $x_2 = \circ(x_1, x_1) = x_1x_1$

Für  $a^{2^n}$ ,  $n \geq 1$ , ist dann als Startregel  $x_1 = a^{2^{n-1}}$  zu verwenden (oder als Schlußregel  $t = x_2 = x_1^{2^{n-1}}$ ).

Bei dieser einfachen Grammatik ist die Richtigkeit der Regeln offensichtlich – sie wirken geradezu wie eine andere Schreibweise der Makro-Produktionen – wie es bei komplexeren Grammatiken aussieht, betrachten wir im folgenden Beispiel:

**Beispiel 7.2** (Example 2.7. aus [Fisch68], Seite 133)

Gegeben sei die IO-Makrogrammatik  $G_M$  mit den Produktionen  $S \rightarrow F(1)$ ,  $F(x) \rightarrow G(F(x1))$ ,  $F(x) \rightarrow G(x)$  und  $G(x) = xcx$ .

Man betrachte die wie im vorhergehenden Beispiel die termgenerierende IO-Makrogrammatik, die die „passenden“ Terme erzeugt, und wende die Konstruktion aus

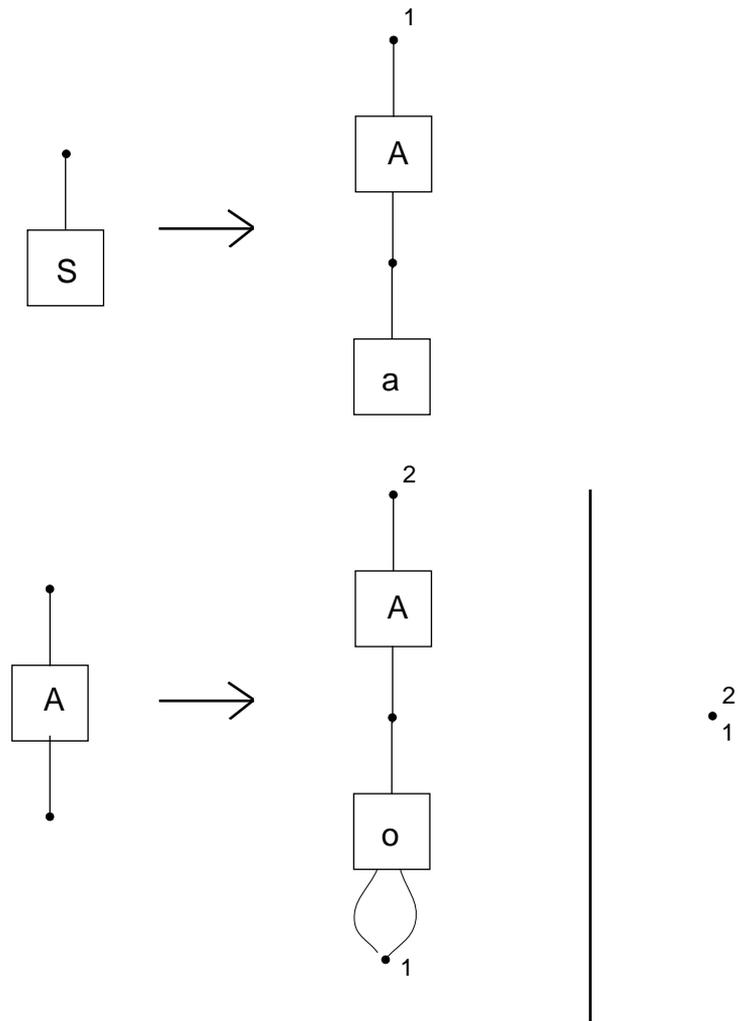
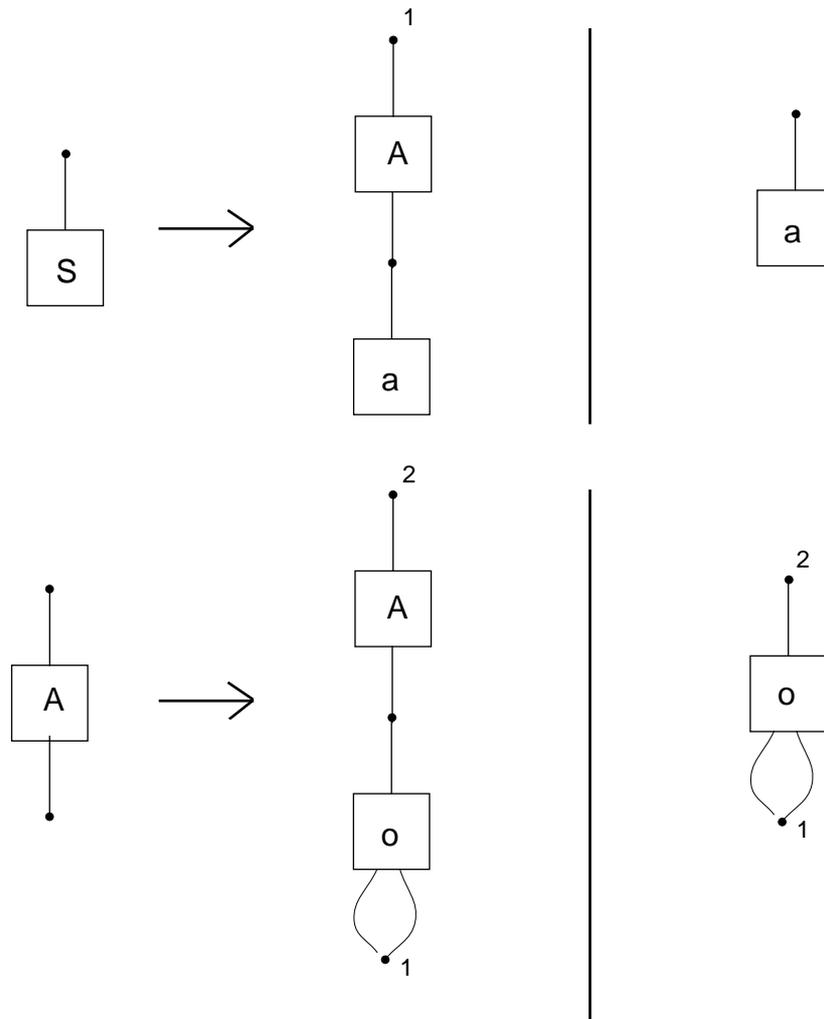


Abbildung 7.2: Produktionen aus  $G$

Abbildung 7.3: Produktionen aus  $G$  nach Beseitigung der  $\epsilon$ -Produktion

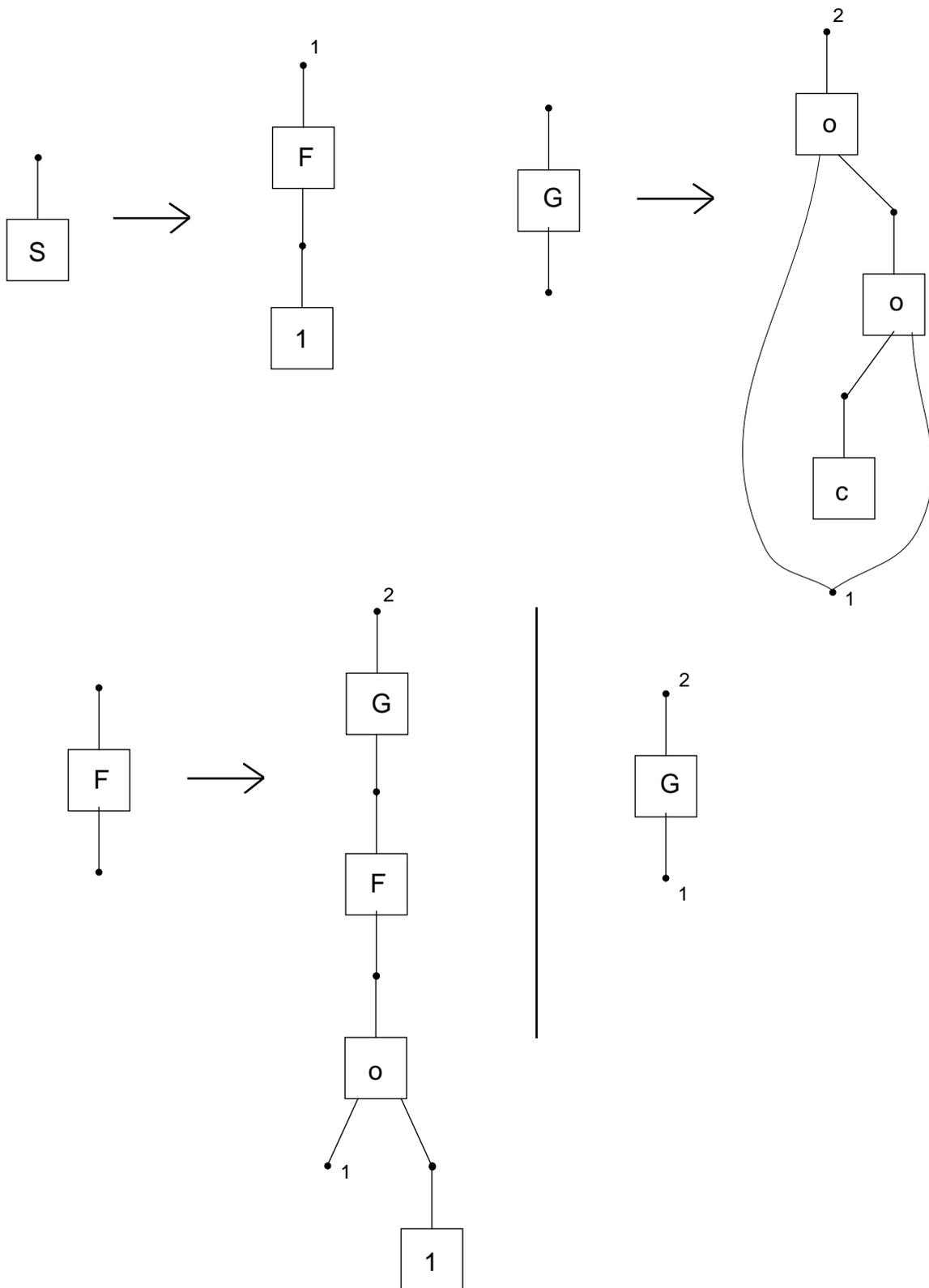
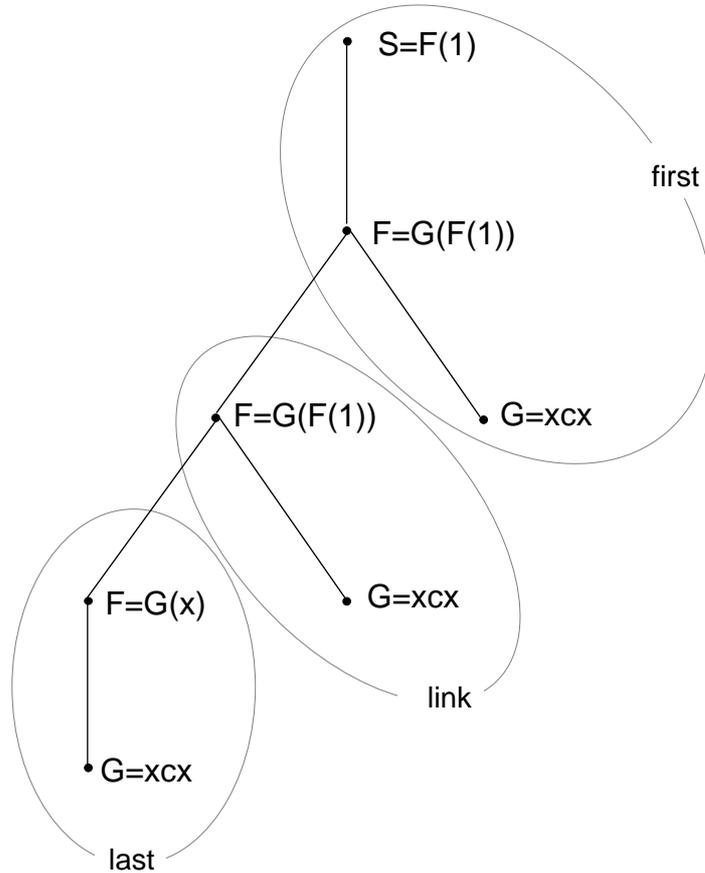


Abbildung 7.4: Produktionen zu  $G_2$

Abbildung 7.5: Ableitungsbaum für  $1^3(c1^3)^7$ 

Abschnitt 6.1 an. So erhält man die Hypergraph-Grammatik  $G_2$  mit den Produktionsregeln aus Abbildung 7.4 .

Gesucht seien jetzt die Regeln für  $1^3(c1^3)^7$ . Wir betrachten dazu zunächst den Ableitungsbaum (Abbildung 7.5) und konstruieren anhand dessen *first*, *link* und *last* (Abbildungen 7.6, 7.7 und 7.8 ) mit den zugehörigen Regeln.

Nun untersuchen wir, was sich bei „dreifacher“ Iteration ergibt:

**Start:**  $t = x_2cx_2,$   
 $x_1 = 11.$

**Iteration.1:**  $t = x_2cx_2cx_2cx_2,$   
 $x_1 = 111.$

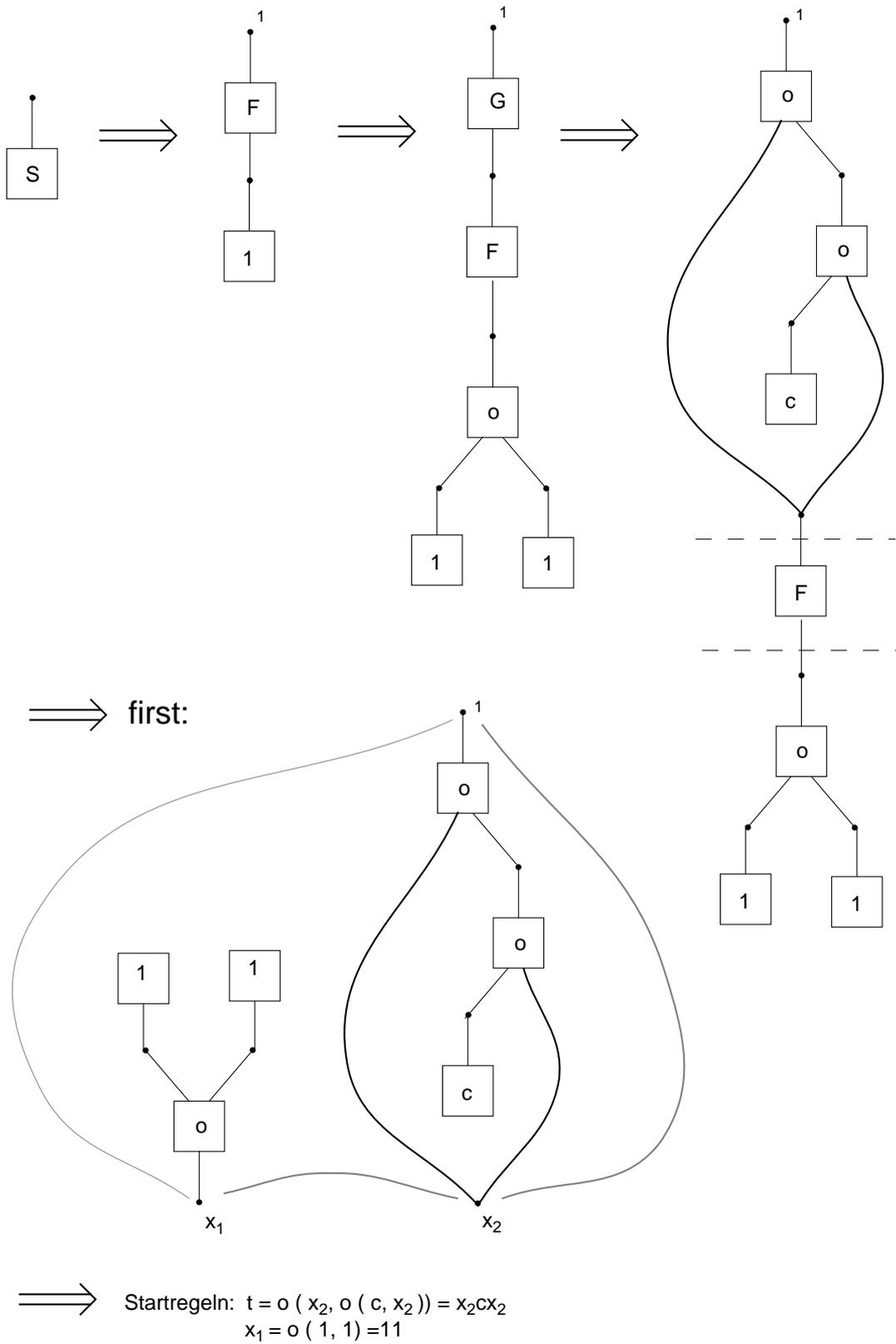


Abbildung 7.6: *first* von „ $1^3(c1^3)^7$ “

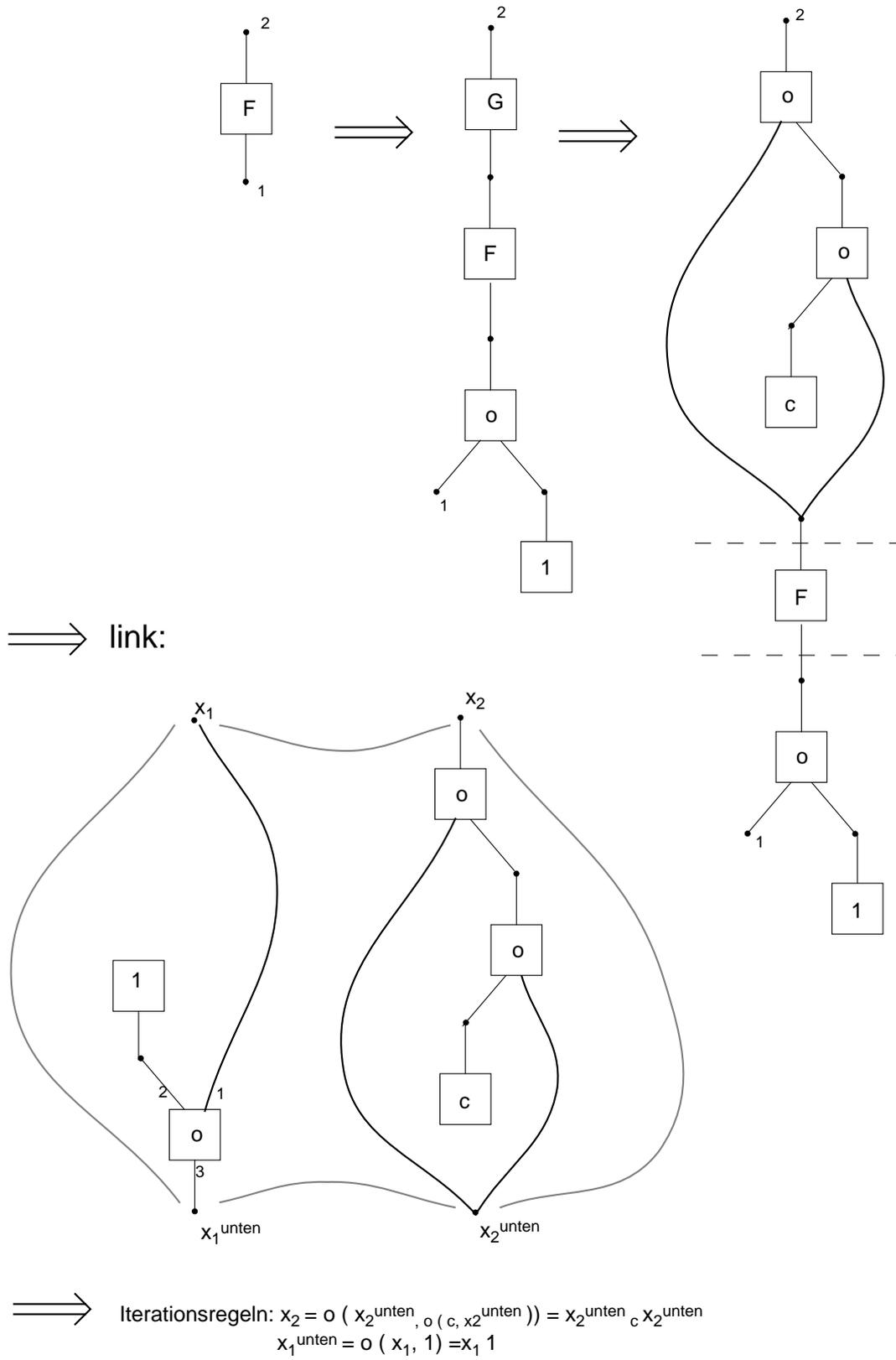
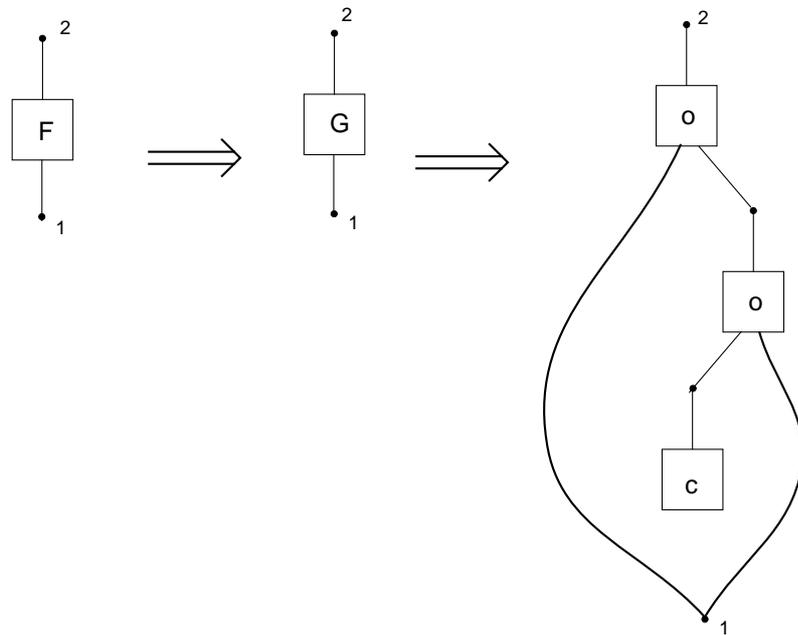
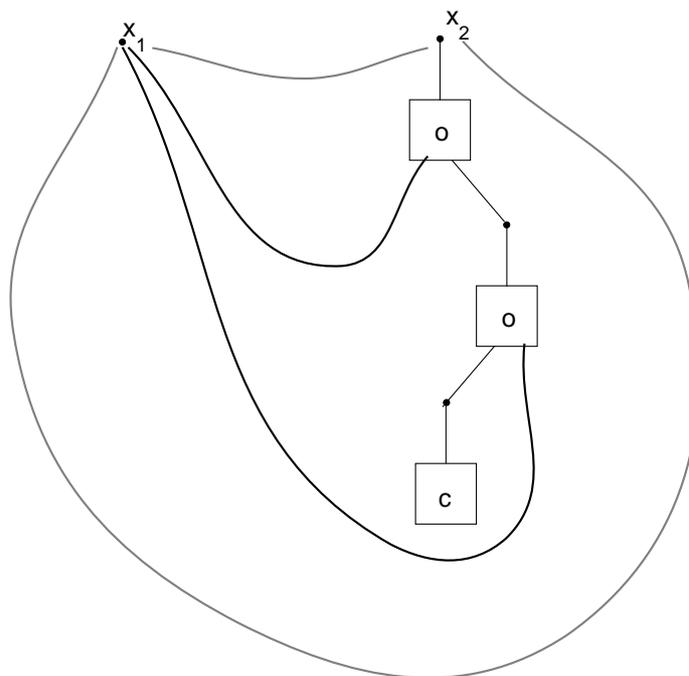


Abbildung 7.7: *link* von „ $1^3(c1^3)^7$ “



⇒ last:



⇒ Schlussregel:  $x_2 = o(x_1, o(c, x_1)) = x_1 c x_1$

Abbildung 7.8: last von „ $1^3(c1^3)^7$ “



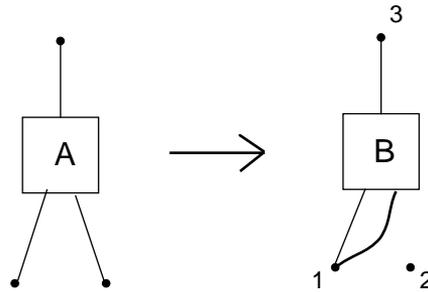


Abbildung 7.9: Problematische Produktion

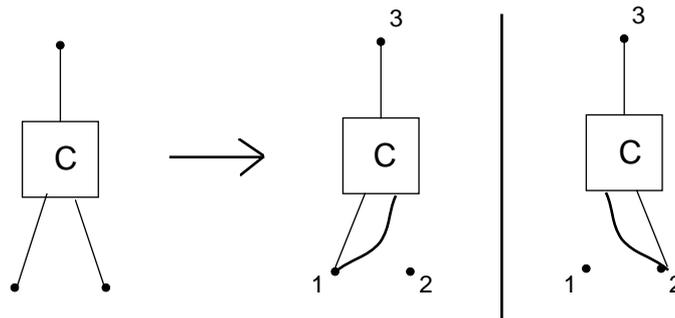


Abbildung 7.10: Noch problematischere Produktionen

### Schritt 1: Beseitigen singularer Produktionen

Da Produktionen Formen wie in Abbildung 7.9, insbesondere auch wie in Abbildung 7.10 haben können, wirkt dieser Schritt etwas „unelegant“. Sei  $A_1, \dots, A_n$  eine beliebige aber feste Aufzählung aller nichtterminalen Label  $(\Sigma - \Delta)$ .

1.  $i := 1$ .
2. Ist  $p_{sing} : A_i \rightarrow A_j$  in  $P$  mit  $i \neq j$  (genauer:  $p_{sing} = (A_i, H_{sing}) \in P$  mit  $|H_{sing}| = |H_{sing}|_N = 1$ ) und  $p = (B, H) \in P$  und tauchen in  $H$   $k$  Hyperkanten mit Label  $A_i$  auf, dann nehme  $p_{neu} = (B, H_{neu})$  mit  $H \Rightarrow_{p_{sing}}^l H_{neu}$ ,  $0 \leq l \leq k$  in  $P_{temp}$  auf – es kommen also zu  $p$   $2^k - 1$  Produktionen hinzu (die Reihenfolge der Ableitung ist beliebig). Dies ist für alle Produktionen der Form  $p_{sing} : A_i \rightarrow A_j$  durchzuführen und jeweils auf alle Produktionen  $p \in P$  anzuwenden – außer auf die für dieses  $p$  gerade hinzugefügten Produktionen (das wäre doppelte Arbeit)

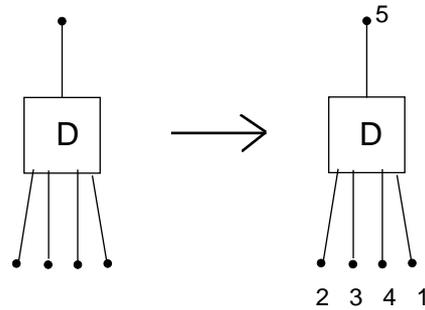


Abbildung 7.11: Singuläre Produktion (Nummerierung der unteren externen Knoten beachten !)

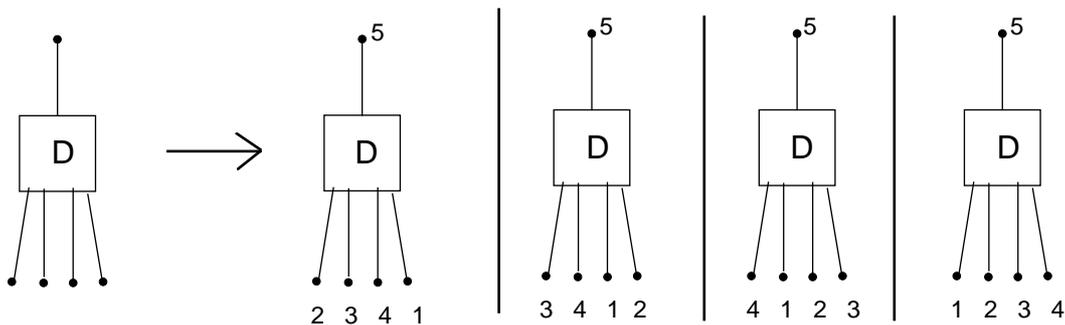


Abbildung 7.12: Zu der Produktion aus Abbildung 7.11 konstruierte Produktionen

3. (der Übersichtlichkeit halber in einem eigenen Schritt)  
 Für die Menge  $M_i$  der Produktionen der Form  $A_i \rightarrow A_i$  bestimme man die „transitive Hülle“:  
 Auf jede rechte Seite einer Produktion aus  $M_i$  wende man jede Produktion aus  $M_i$  an. Entsteht dabei ein Hypergraph  $H$ , der nicht rechte Seite einer Produktion aus  $M_i$  ist, dann ergänze man  $M_i$  um die Produktion  $A_i \rightarrow H$ . Dies ist zu wiederholen, bis keine Änderung mehr auftritt.  
 Für die Produktion aus Abbildung 7.11 z. B. ergeben sich dann genau die Produktionen aus Abbildung 7.12. Hinweis: Die Anzahl der Produktionen mit Dschungel-Eigenschaft der Form  $A_i \rightarrow A_i$  ist immer endlich: Sei  $n = \text{rank}(A_i)$ , dann kann es nicht mehr als  $(n - 1)^{n-1}$  Produktionen geben – das  $n$ -te Tentakel ist immer als einziges mit dem  $n$ -ten externen Knoten verbunden.  
 Für die so erhaltenen (singulären) Produktionen führe man die gleichen Operationen wie unter 2. aus – wobei auf die Anwendung auf eine Produktion aus  $M_i$  natürlich verzichtet werden kann.
4. Streiche alle Produktionen der Form  $A_i \rightarrow A_j$  aus  $P_{temp}$ .
5.  $i := i + 1$ .

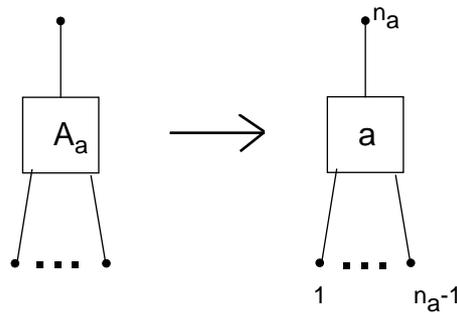


Abbildung 7.13: Einführung neuer terminaler Produktionen

6. Falls  $i \leq n$  ist, gehe zu 2.,  
sonst fertig.

### Schritt 2: Terminale Produktionen einführen

Für jedes  $a \in \Delta$ ,  $\text{rank}_\Sigma(a) = n_a$ , sei  $A_a$  ein neues Zeichen, ergänze  $P_{temp}$  um die Produktion aus Abbildung 7.13.

### Schritt 3: Beseitigen der terminalen Hyperkanten in den übrigen Produktionen

Siehe Abbildung 7.14.

### Schritt 4: Kontraktion

Ist  $p = (B, H) \in P_{temp}$  und  $|H| \geq 3$ , dann ersetze diese Produktion durch zwei andere mit kleineren rechten Seiten wie in Abbildung 7.15. Dabei ist zu beachten, daß der Knoten  $v$  mit keiner weiteren Hyperkante verbunden ist – die Verbindung mit  $e_2$  ist allerdings nicht notwendig (das Verfahren funktioniert also auch, wenn die Hypergraph-Grammatik nicht die reine-Dschungel-Eigenschaft besitzt).

Ein solcher Knoten läßt sich immer (in einer rechten Seite mit mindestens zwei Hyperkanten) finden: Betrachte den Knoten, dessen Eingangskante das zweithöchste Element einer Einbettung der  $v$ -Ordnung ist.

Dieser Schritt ist zu wiederholen, bis keine rechte Seite mehr als zwei Hyperkanten besitzt.

### Schritt 5: Schluss

Setze  $G_{CNF} := G_{temp}$ .

**Folgerung 7.4** Zu jeder kontextfreien Hypergraph-Grammatik  $G$  mit Dschungel-Eigenschaft gibt es eine Hypergraph-Grammatik  $G_{CNF}$  mit Dschungel-Eigenschaft in Chomsky-Normalform mit  $L(G_{CNF}) = L(G)$ .

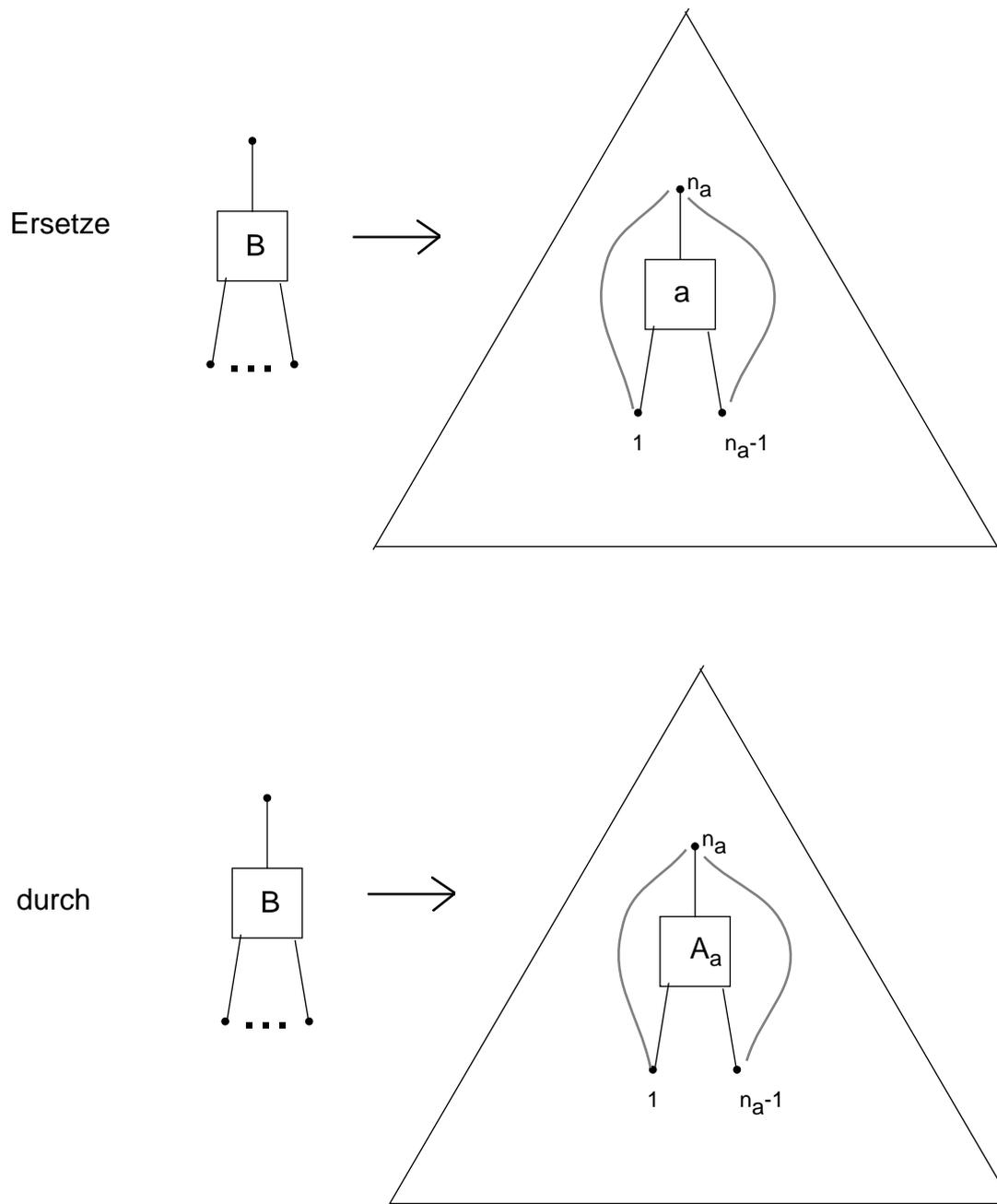


Abbildung 7.14: Beseitigung terminaler Hyperkanten in den übrigen Produktionen

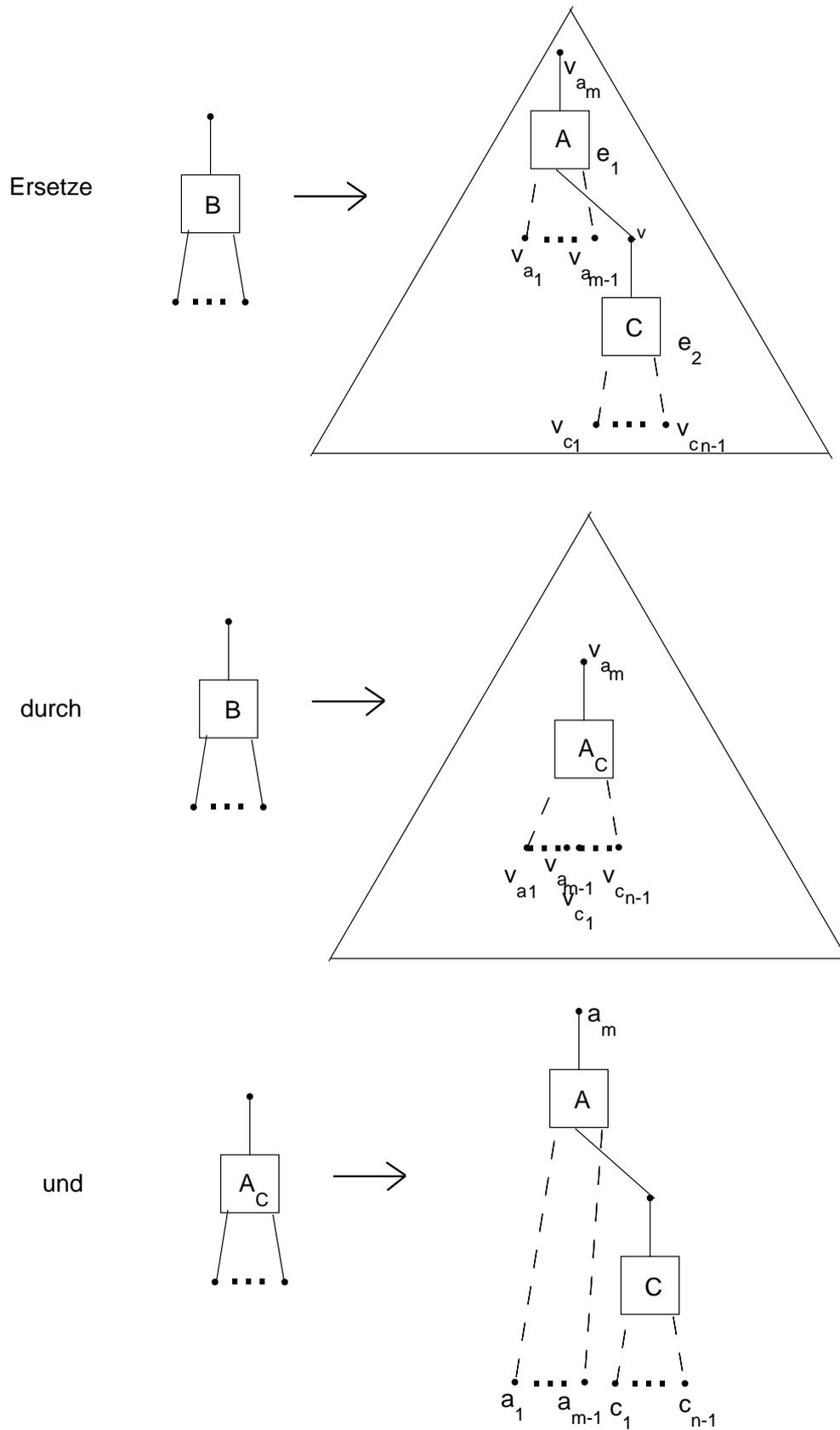


Abbildung 7.15: Kontraktion

## 7.2.2 Greibach-Normalform

Die Greibach-Normalform für kontextfreie Grammatiken sieht so aus, daß auf der rechten Seite einer Produktion immer ein terminales Zeichen gefolgt von einer Anzahl nichtterminaler Zeichen steht. Es stellt sich nun die Frage, wie dies auf kontextfreie Hypergraph-Grammatiken mit Dschungel-Eigenschaft zu übertragen ist – ohne Dschungel-Eigenschaft ist diese Frage u. a. in [Engel92] und [EnHeLe] behandelt worden.

Die naheliegende Antwort wäre vielleicht die, daß auf der rechten Seite einer Produktion die Hyperkante mit terminalem Label sich „ganz oben“ befindet – also Eingangskante des höchsten externen Knoten ist. Dies läßt sich aber nicht realisieren wie man sich anhand der Sprache aus Beispiel 4.3 überlegen kann:

Man müßte als erstes die „Weiche“ (die Hyperkante mit dem Label „+“) produzieren und – da alle Satzformen Dschungel sein sollen – für die beiden „Gleise“ je eine Hyperkante mit nichtterminalem Label. Dies führt aber zum „klassischen Problem“ der Kontextfreiheit – man kann nicht mehr die Längen der Gleise vergleichen und deswegen nicht gewährleisten, daß diese gleichlang sind.

Wir können aber erreichen, daß „ganz unten“ auf der rechten Seite einer Produktion immer eine Hyperkante mit terminalem Label steht und „darüber“ kettenförmig eine Anzahl von Hyperkanten mit nichtterminalem Label. Die rechte Seite sehe wie in Abbildung 7.16 skizziert aus.

**Definition 7.5 (Greibach-Normalform)** *Die kontextfreie Hypergraph-Grammatik  $G = (\Sigma, \Delta, P, S)$  mit reiner Dschungel-Eigenschaft befindet sich in Greibach-Normalform, wenn für jede Produktion  $p = (B, H) \in P$  gilt, daß*

1. *zwischen allen Hyperkanten aus  $H$  die  $v$ -Ordnung definiert ist (Kettenförmigkeit) und*
2. *die einzige Hyperkante aus  $H$  mit terminalem Label das kleinste Element der  $v$ -Ordnung ist.*

Es soll nun beschrieben werden, wie man zu einer kontextfreien Hypergraph-Grammatik  $G$  mit reiner Dschungel-Eigenschaft eine Hypergraph-Grammatik  $G_{GNF}$  in Greibach-Normalform mit  $L(G_{GNF}) = L(G)$  konstruieren kann. Da wir zuvor schon gezeigt haben, daß es eine Hypergraph-Grammatik  $G_{CNF}$  in Chomsky-Normalform mit  $L(G_{CNF}) = L(G)$  gibt, bedienen wir uns dieser, um daraus eine Hypergraph-Grammatik mit den gewünschten Eigenschaften zu gewinnen:

### Konstruktionsanleitung:

Sei  $G_{CNF} = (\Sigma_{CNF}, \Delta_{CNF}, P_{CNF}, S_{CNF})$  eine kontextfreie Hypergraph-Grammatik mit reiner Dschungel-Eigenschaft in Chomsky-Normalform.

Setze  $G_{temp} := G_{CNF}$  und bearbeite  $G_{temp}$  wie folgt:

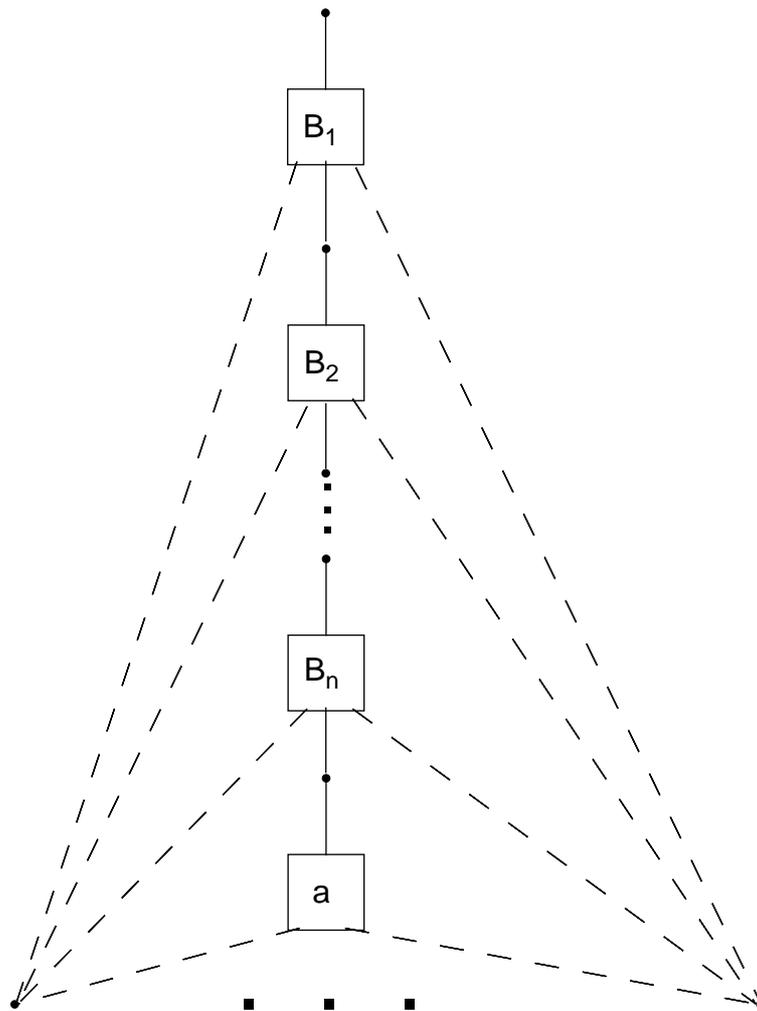
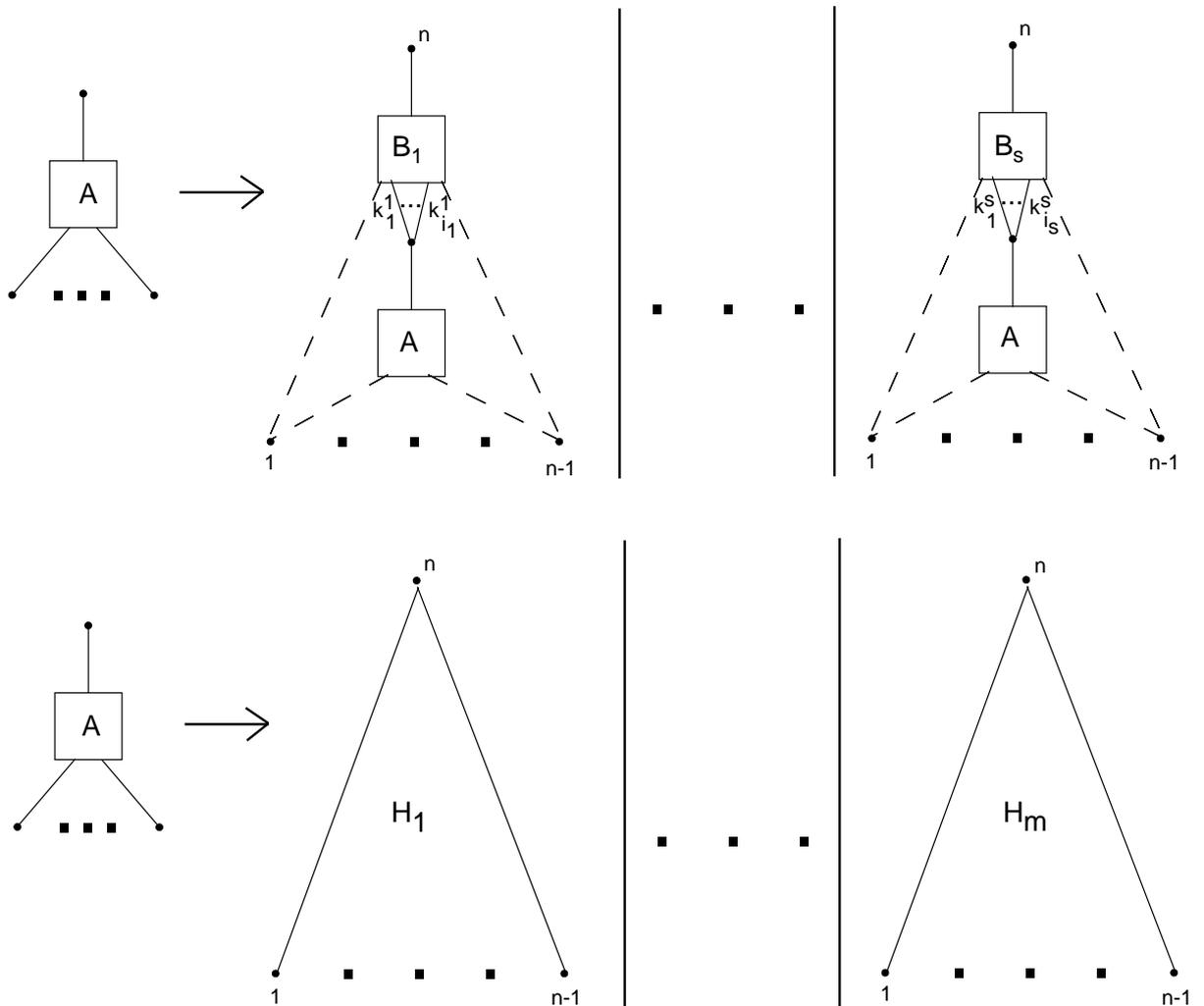


Abbildung 7.16: „Muster“ einer rechten Seite in Greibach-Normalform

Abbildung 7.17: Alle Produktionen mit  $A$  auf der linken Seite

**Schritt 1: Beseitigen von „Links“-Rekursion** Sei  $A \in \Sigma_{temp} - \Delta_{temp}$  mit  $rank(A) = n$  und seien die in Abbildung 7.17 skizzierten Produktionen alle mit  $A$  auf der linken Seite. In den  $H_i$  ( $1 \leq i \leq m$ ) tauche das  $A$  nicht als Label der „kleinsten“ Hyperkante auf. Die gestrichelten Linien stehen für eine „Menge von Tentakeln“ – von  $B_1$  aus z. B. gehen noch  $rank(B_1) - 1 - k_{i_1}^1$  Tentakel zu den ersten  $n - 1$  externen Knoten, von  $A$  aus sind es  $n - 1$  Tentakel. Dann werden diese Produktionen ersetzt durch die in den Abbildungen 7.18 und 7.19 skizzierten Produktionen.  $C$  sei ein neues Symbol mit  $rank(C) = n + 1 = rank(A) + 1$  und wird benutzt, um die externen Knoten ( $n$  an der Zahl) und den Knoten mit  $A$  als Eingangskante „festzuhalten“.

**Schritt 2:** Wähle Aufzählung  $D_1, \dots, D_m$  aller nichtterminalen Symbole aus  $\Sigma_{temp}$

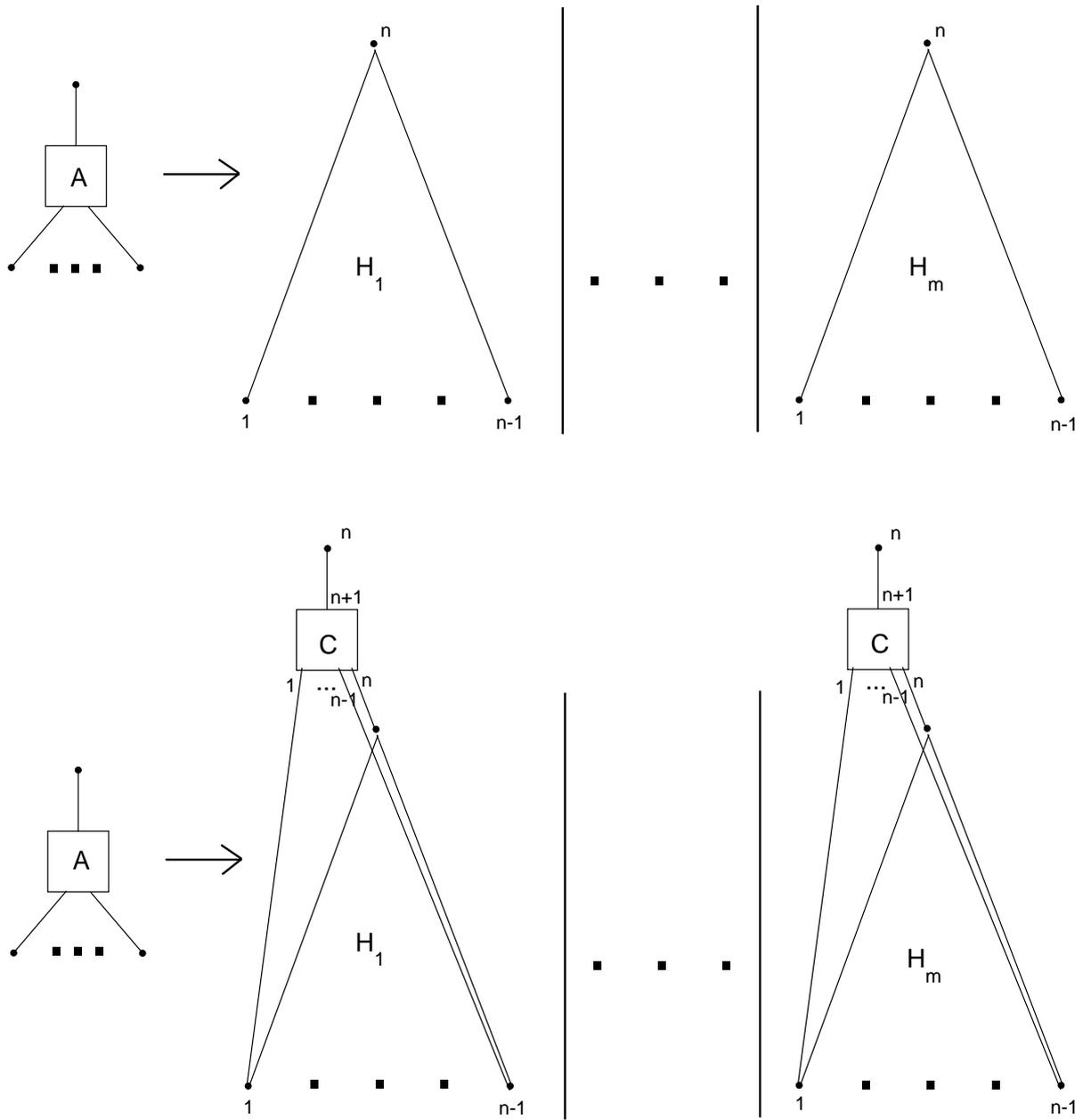


Abbildung 7.18: ... werden ersetzt durch diese und ...

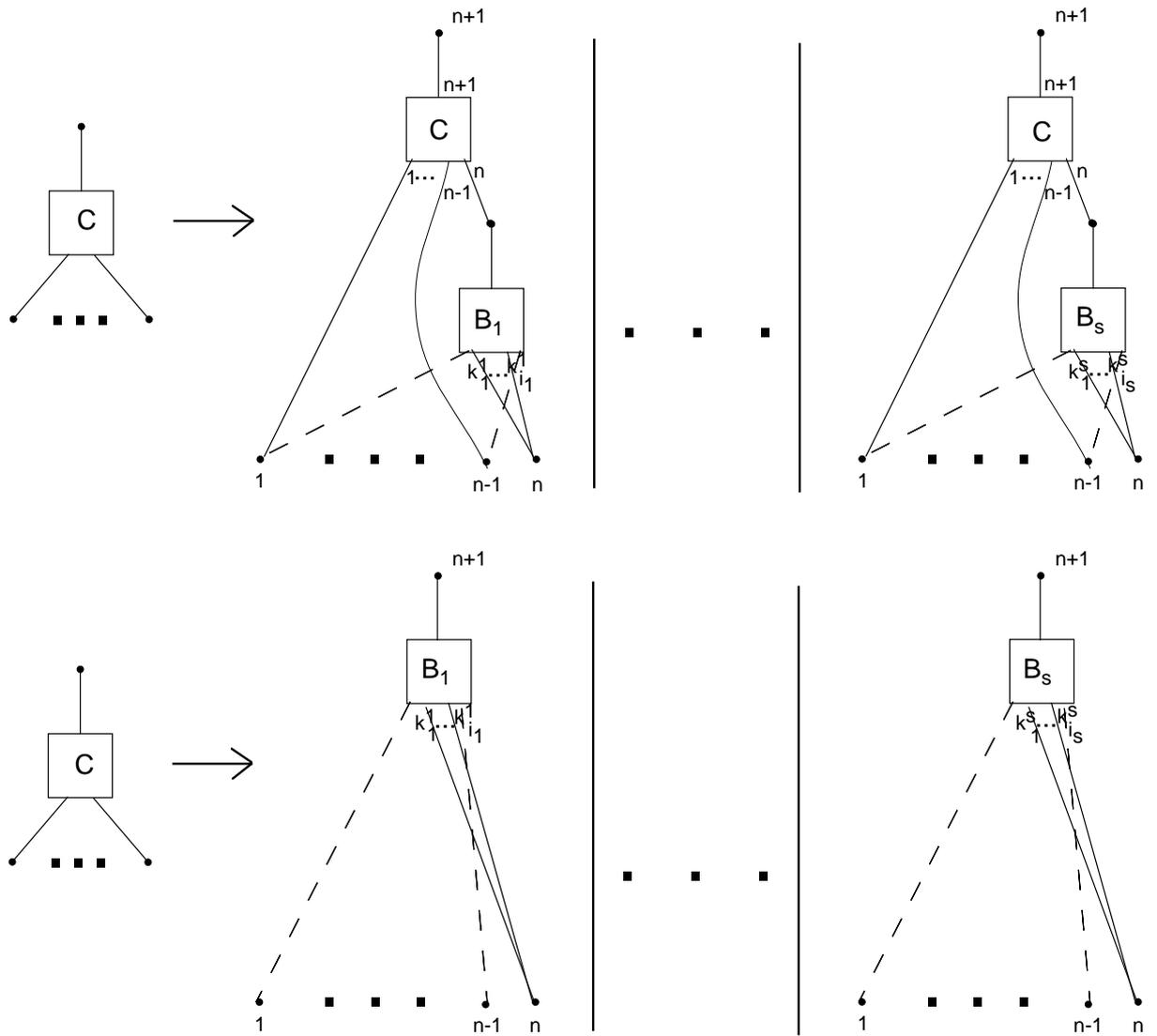
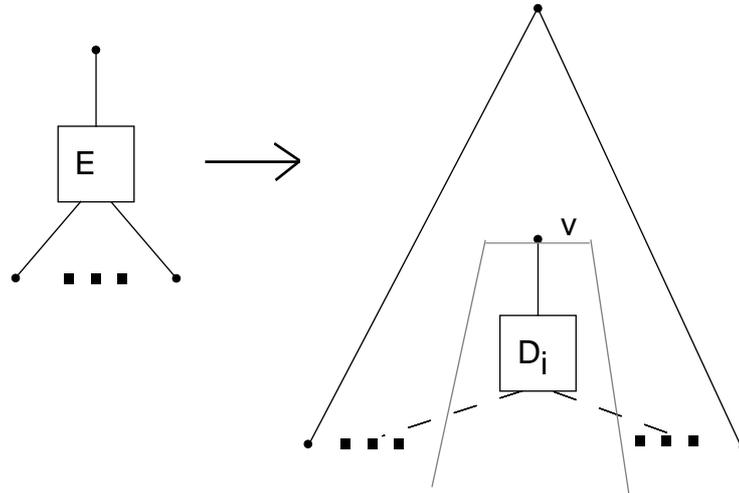


Abbildung 7.19: ... diese Produktionen

Abbildung 7.20: Hier ist  $D_i$  ganz unten auf der rechten Seite

– aufgrund von Schritt 1 sind dies im Extremfall doppelt so viele wie in  $\Sigma_{CNF}$ .

1.  $i := 1$ .
2. Ist  $p = (E, H_p) \in P$  eine Produktion wie in Abbildung 7.20 skizziert (die mit  $D_i$  markierte Hyperkante befinde sich „ganz unten“ in  $H_p$ ) und sind die Produktionen, die in Abbildung 7.21 skizziert sind, genau diejenigen mit  $D_i$  auf der linken Seite, dann ersetze  $p$  durch die in Abbildung 7.22 skizzierten Produktionen.  
Zur Erläuterung: Gilt z. B. in  $H_s$  aus Abbildung 7.21 für eine Hyperkante  $e$ :  $nod(e, j) = ext(l)$  und in Abbildung 7.20 (sei  $e_{D_i}$  die gesondert gezeichnete Hyperkante aus Abbildung 7.20):  $nod(e_{D_i}, l) = q$ , dann gelte in Abbildung 7.22:  $nod(e, j) = ext(q)$ .
3. Beseitige Linksrekursion (wie in Schritt 1 – wichtig sind insbesondere die mit  $D_{i+1}$ ).
4. Falls durch das Beseitigen der Linksrekursion eine Produktion entsteht, auf deren rechter Seite ganz unten eine mit  $D_j$ ,  $j \leq i$ , markierte Hyperkante steht, dann wende auf diese Hyperkante alle Produktionen mit  $D_j$  auf der linken Seite an und ersetze diese Produktion durch die so erhaltene Menge von Produktionen.
5.  $i := i + 1$ .
6. Falls  $i \leq n$  gehe zu 2., sonst fertig.

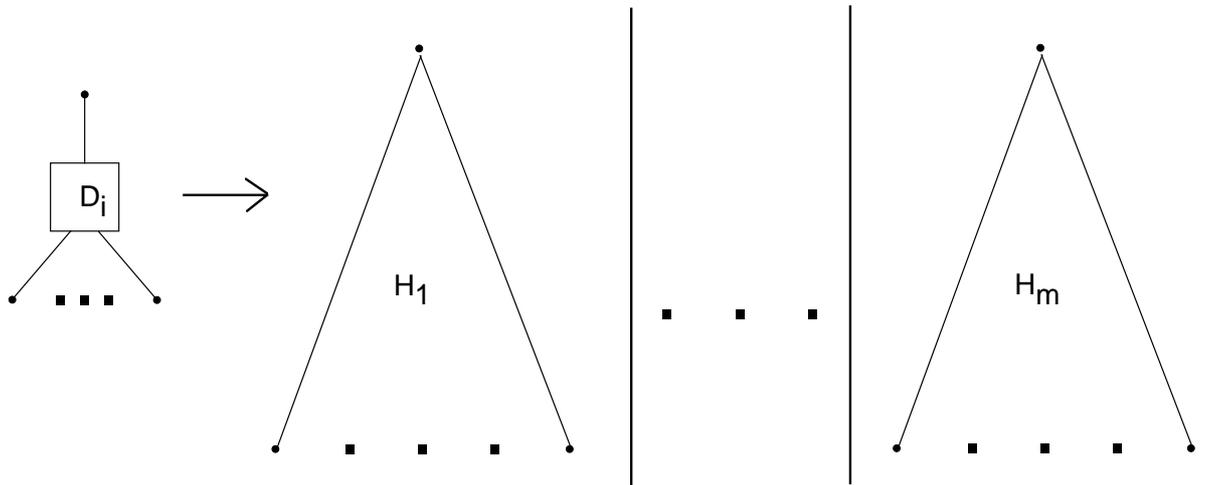
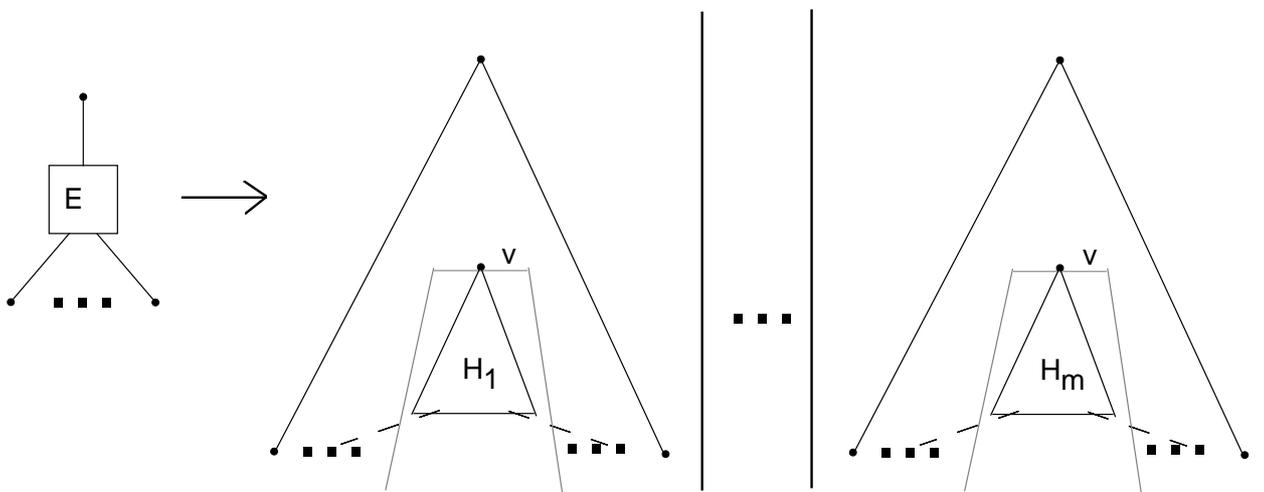
Abbildung 7.21: Alle Produktionen mit  $D_i$  auf der linken Seite

Abbildung 7.22: Ersatz für die Produktion aus Abbildung 7.20

**Schritt 3: Schluss**

Setze  $G_{GNF} := G_{temp}$ .

**Folgerung 7.6** *Zu jeder kontextfreien Hypergraph-Grammatik  $G$  mit Dschungel-Eigenschaft gibt es eine Hypergraph-Grammatik  $G_{GNF}$  in Greibach-Normalform mit  $L(G_{GNF}) = L(G)$ .*

Mit den Normalformen ist dann auch das „Wort-Problem“ für kontextfreie Hypergraph-Grammatiken mit Dschungel-Eigenschaft zu beantworten:

**Folgerung 7.7** *Für einen Hypergraphen  $H$  und eine kontextfreie Hypergraph-Grammatik  $G$  mit Dschungel-Eigenschaft ist entscheidbar, ob  $H \in L(G)$  ist.*

### 7.3 Ausblick

Die vorliegende Arbeit liefert eine Reihe von interessanten Ansatzpunkten für weitere Untersuchungen.

So ergibt sich z. B. aus den in den Kapiteln 5 und 6 vorgestellten Konstruktionsverfahren jeweils eine große Nähe zwischen vorgegebener und konstruierter Grammatik. Dies läßt hoffen, daß aus Erkenntnissen, die für die eine Grammatik vorliegen, neue für die andere gewonnen werden können — wie dies z. B. im Unterkapitel 7.1 praktiziert wurde.

Oder es kann untersucht werden, welche L-attribuierten Grammatiken und IO-Makrogrammatiken den verschiedenen linearen Hypergraph-Grammatiken entsprechen. Vermutlich ergeben sich die folgenden Zusammenhänge (Linearität sei „wie üblich“ definiert):

- untenlinear:
  - die nichtterminalen Funktionssymbole der IO-Makrogrammatik stehen „ganz innen“, d. h. sie haben keine Argumente ( $\rho(F) = 0$  für  $F \in \mathcal{F}$ ).
  - die nichtterminalen Symbole der L-attribuierten Grammatik besitzen keine inheriten Attribute.
- obenlinear:
  - die nichtterminalen Funktionssymbole der IO-Makrogrammatik stehen „ganz außen“, d.h. sie treten nicht als Argumente auf.
  - falls auf der rechten Seite ein nichtterminales Symbol steht ( $A \rightarrow \dots B \dots$ ), wird das synthetische Attribut des linksseitigen Symbols gleich dem des rechtsseitigen gesetzt ( $Syn_A = Syn_B$ ).

Vielversprechend scheint auch die Suche nach Möglichkeiten zur Analyse von Dschungeln, die sich über die vorgestellte Greibach-Normalform ergeben. Hierzu muß die Normalform noch verschärft werden, so daß für nicht terminale Produktionen gilt, daß der „unterste“ interne Knoten mit dem ersten Tentakel der „untersten“ nichtterminalen Hyperkante verbunden ist. Dann können vermutlich Verfahren, wie sie für LL(1)-Grammatiken existieren, in ähnlicher Art und Weise für Hypergraph-Grammatiken mit den entsprechenden weiteren Eigenschaften (LL(1)-Bedingungen) eingesetzt werden.

Zuletzt sei noch auf jene Aspekte verwiesen, die sich einerseits aus der Bedeutung L-attribuiertter Grammatiken für Programmiersprachen und andererseits aus der guten Sichtbarkeit der Unabhängigkeit einzelner Teile des Dschungels voneinander ergeben. Hier lassen sich eventuell Möglichkeiten zur Parallelisierung bei der Codeerzeugung automatisch erkennen und — in einer Zeit, wo die Kosten für einzelne Prozessoren relativ niedrig sind — nutzen.

# Literaturverzeichnis

- [CoFr1] B. Courcelle, P. Franchi-Zannettacci: Attribute Grammars and Recursive Program Schemes I. *Theoretical Computer Science* **17**, 163-191 (1982)
- [CoFr2] B. Courcelle, P. Franchi-Zannettacci: Attribute Grammars and Recursive Program Schemes II. *Theoretical Computer Science* **17**, 235-257 (1982)
- [DuPaSeSp] J. Duske, R. Parchmann, M. Sedello, J. Specht: IO-Macrolanguages and Attributed Translations. *Information and Control* **35**, 87-105 (1977)
- [DuPaSp] J. Duske, R. Parchmann, J. Specht: Szilard Languages of IO-Grammars. *Information and Control* **40**, 319-331 (1979)
- [Engel92] J. Engelfriet: A Greibach Normal Form for Context-free Graph Grammars. In: W. Kuich (Ed.) *Automata Languages and Programming (Lecture Notes in Computer Science 623)*, 138-149. Springer-Verlag, Berlin, Heidelberg, New York (1992)
- [EnHe91] J. Engelfriet, L. M. Heyker: The string generating power of context-free hypergraph grammars. *Journal of Computer System Science* **43**, 328-360 (1991)
- [EnHe92] J. Engelfriet, L. M. Heyker: Context-free hypergraph grammars have the same term-generating power as attribute grammars. *Acta Informatica* **29**, 161-210 (1992)
- [EnHeLe] J. Engelfriet, L. M. Heyker, G. Leih: Context-free graph languages of bounded degree are generated by apex graph grammars. *Acta Informatica* **31**, 341-378 (1994)
- [EnLeRo] J. Engelfriet, G. Leih, G. Rozenberg: Apex Graph Grammars and Attribute Grammars. *Acta Informatica* **25**, 537-571 (1988)
- [EnLe89] J. Engelfriet, G. Leih: Linear Graph Grammars: Power and Complexity. *Information and Computation* **81**, 88-121 (1989)
- [Feder71] J. Feder: Plex Languages. *Information Sciences* **3**, 225-241 (1971)

- [Fisch68] M. J. Fischer: Grammars with Macro-Like Productions. IEEE 9th Annual Symposium on Switching and Automata Theory, 131-142 (1968)
- [Hab92] A. Habel: Hyperdege Replacement: Grammars and Languages. Lecture Notes in Computer Science **643**. Springer-Verlag, Berlin 1992.
- [Har78] M. A. Harrison: Introduction to Formal Language Theory. Addison-Wesley Publishing Co., Reading, Massachusetts, 1978.
- [HoPl88] B. Hoffmann, D. Plump: Jungle Evaluation for Efficient Term Rewriting. In: J. Grabowski, P. Lescanne, W. Welcher (eds.) Algebraic and logic programming (Lecture Notes in Computer Science **343**), 191-203. Springer-Verlag, Berlin, Heidelberg, New York (1988)
- [Hüm93] C. Hümpel: Über echte Teilklassen von indizierten Sprachen und ihre Einbettung in die Chomsky-Hierarchie. Dissertation, Hannover 1993.

# Lebenslauf

- 02.09.1964 geboren in Hünzingen als Kind meiner Eltern Helmut Ehlermann und Ursel von Fintel, geb. Helmke, verw. Ehlermann
- 1971 - 1974 Besuch der Grundschule in Hünzingen  
1974 - 1975 Besuch der Marktschule in Walsrode  
1975 - 1977 Besuch der Orientierungsstufe Walsrode  
1977 - Juni 1984 Besuch des Gymnasiums Walsrode  
19. Juni 1984 Abitur
- Juli 1984 - Sept. 1985 Grundwehrdienst in Munster
- Okt. 1985 - März 1992 Studium an der Technischen Universität Carolo-Wilhemina zu Braunschweig  
24. März 1992 Studienabschluß als Diplom-Informatiker
- April 1992 - Juli 1992 Wissenschaftlicher Angestellter des Instituts für theoretische Informatik der Technischen Universität Carolo-Wilhemina zu Braunschweig
- Juni 1993 - Juni 1997 Wissenschaftlicher Angestellter des Instituts für Informatik der Universität Hannover