

# 11. Hierarchie der LA-Grammatik

## 11.1 Generative Kapazität der unbeschränkten LAG

### 11.1.1 Grundeigenschaft der unbeschränkten LAG

Unrestricted LA-grammar accepts and generates all and only the recursive languages.

### 11.1.2 Theorem 1

Die LA-Grammatik in ihrer unbeschränkten Grundform analysiert und generiert *nur* die rekursiven Sprachen.

*Beweis:* Gegeben sei eine Eingabekette von endlicher Länge  $n$ . Jedes Wort in dieser Eingabekette hat endlich viele lexikalische Lesarten ( $> 0$ ).

*Ableitungslänge 1:* Die endliche Menge der Anfangszustände  $ST_S$  und alle Lesarten des ersten Wortes  $w_1$  ergeben eine endliche Menge wohlgeformter Ausdrücke  $WE_1 = \{(ss' rp_S) \mid ss' \in (W^* \times C^+)\}$ .

*Kombination  $n$ :* Kombination  $k - 1$ ,  $k > 1$ , hat eine endliche Menge wohlgeformter Ausdrücke

$WE_k = \{(ss' rp_i) \mid i \in n, ss' \in (W^* \times C^+), \text{ und die Oberfläche von } ss' \text{ hat Länge } k\}$   
hervorgebracht. Das nächste Wort  $w_{k+1}$  hat endlich viele Lesarten.

Deshalb ist das kartesische Produkt aller Elemente von  $WE_k$  und aller Lesarten des nächsten Wortes  $w_{k+1}$  eine endliche Menge von geordneten Paaren. Jedes Paar enthält ein Regelpaket mit endlich vielen Regeln. Deshalb ergibt Kombination  $k$  nur endlich viele neue Satzanfänge. Die Ableitung dieser endlichen Menge neuer Satzanfänge ist entscheidbar, weil die kategorialen Operationen als totale rekursive Funktionen definiert wurden.

Q.E.D.

### 11.1.3 Theorem 2

Der Formalismus der LA-Grammatik analysiert und generiert *alle* rekursiven Sprachen.

*Beweis:*  $L$  sei eine rekursive Sprache mit der Wortmenge  $W$ . Weil  $L$  rekursiv ist, gibt es eine totale rekursive Funktion  $\varrho: W^* \rightarrow \{0,1\}$ , also die charakteristische Funktion von  $L$ .  $LAG^L$  sei eine LA-Grammatik mit der folgenden Definition:

Die Menge der Wortoberflächen von  $LAG^L$  ist  $W$ .

Die Menge der Kategorie-segmente  $C =_{def} W \cup \{0,1\}$

Für beliebige  $e, f \in W^+$  gilt,  $[e (f)] \in LX$  dann und nur dann wenn  $e = f$ .

$$LX =_{def} \{[a (a)], [b (b)], [c (c)], [d (d)], \dots\}$$

$$ST_S =_{def} \{[(seg_c) \{r_1, r_2\}]\}, \text{ where } seg_c \in \{a, b, c, d, \dots\}$$

$$r_1: (X) (seg_c) \Rightarrow (X seg_c) \quad \{r_1, r_2\}$$

$$r_2: (X) (seg_c) \Rightarrow \varrho (X seg_c) \quad \{ \}$$

$$ST_F =_{def} \{[(1) rp_2]\}$$

Nach jedem Kombinationschritt erlaubt  $rp_1$  zwei mögliche Fortsetzungen: Anwendung von  $r_1$  um die Eingabekette weiterzulesen (und in die Resultatkategorie zu kopieren) oder Anwendung von  $r_2$ , um zu testen, ob die

bisher gelesene Eingabe einen wohlgeformten Ausdruck von  $L$  darstellt. In letzterem Falle wird die Funktion  $\varrho$  auf die bisher angesammelten Kategorie-segmente der Eingabe angewendet (die der Oberfläche der Eingabe gleichen). Wenn das Ergebnis der Anwendung von  $r_2$  in dem Zustand  $[(1) rp_2]$  resultiert, wird die Oberfläche von der Grammatik akzeptiert. Ist das Ergebnis dagegen  $[(0) rp_2]$ , ist die Eingabe kein wohlgeformter Ausdruck von  $L$ .

Da die kategorialen Operationen von  $LAG^L$  jede totale rekursive Funktion sein kann, kann  $LAG^L$  auf  $\varrho$ , also der charakteristischen Funktion von  $L$ , basieren. Deshalb akzeptiert und generiert  $LAG^L$  alle rekursiven Sprachen.

Q.E.D.

#### 11.1.4 Definition der Klasse der A-LAGs.

Die Klasse der A-LAGs besteht aus unbeschränkten LA-Grammatiken und generiert *alle* rekursiven Sprachen.

## 11.2 LA-Hierarchie der A-, B- und C-LAGs

### 11.2.1 Parameter der Komplexität

- Der *Aufwand* an Berechnung pro Regelanwendung, der im schlimmsten Fall benötigt wird.
- Die *Anzahl* der Regelanwendungen, abhängig von der Länge der Eingabe, die bei einer Ableitung im schlimmsten Fall benötigt werden.

### 11.2.2 Hauptansatzpunkte zur Beschränkung von LAGs

*R1*: Beschränkungen der Form kategorialer Operationen, um den möglichen Rechenaufwand beliebiger Regelanwendungen zu begrenzen.

*R2*: Beschränkungen des Ambiguitätsgrades, um die Anzahl der möglichen Regelanwendungen zu begrenzen.

### 11.2.3 Mögliche Einschränkungen kategorialer Operationen

*R1.1*: Beschränkung der Kategorielänge.

*R1.2*: Beschränkung der Kategoriemuster, die bei der Definition kategorialer Operationen verwendet werden dürfen.

### 11.2.4 Definition der Klasse der B-LAGs

Die Klasse der B-LAGs besteht aus LA-Grammatiken, für die eine Konstante  $B$  existiert, so daß bei beliebigen Eingaben der Länge  $n$  die Kategorien der Ableitungszwischenschritte höchstens die Länge  $B \cdot n$  haben.

### 11.2.5 Konstante kategoriale Operationen

$$r_i: (\text{seg}_1 \dots \text{seg}_k \text{ X}) \text{ cat}_2 \Rightarrow \text{cat}_3 \text{ rp}_i$$

$$r_i: (\text{X seg}_1 \dots \text{seg}_k) \text{ cat}_2 \Rightarrow \text{cat}_3 \text{ rp}_i$$

$$r_i: (\text{seg}_1 \dots \text{seg}_m \text{ X seg}_{m+1} \dots \text{seg}_k) \text{ cat}_2 \Rightarrow \text{cat}_3 \text{ rp}_i$$

### 11.2.6 Nichtkonstante kategoriale Operation

$$r_i: (\text{X seg}_1 \dots \text{seg}_k \text{ Y}) \text{ cat}_2 \Rightarrow \text{cat}_3 \text{ rp}_i$$

### 11.2.7 Definition der Klasse der C-LAGs

Die Klasse der C-LAGs besteht aus B-LAGs, für die eine endliche Konstante  $C$  existiert, so daß keine kategoriale Operation  $\text{co}_i$  mehr als  $C$  Segmente in den Satzanfangskategorien überprüft.

### 11.2.8 Die Hierarchie der A-LAGs, B-LAGs, und C-LAGs

Die Klasse der A-LAGs analysiert und generiert alle rekursiven Sprachen, die Klasse der B-LAGs analysiert und generiert alle kontextsensitiven Sprachen, und die Klasse der C-LAGs analysiert und generiert viele kontextsensitive Sprachen, alle kontextfreien Sprachen und alle regulären Sprachen.

## 11.3 Ambiguität in der LA-Grammatik

### 11.3.1 Faktoren für die Anzahl der Regelanwendungen

1. Länge der Eingabe.
2. Zahl der Regeln in einem Regelpaket, das bei einer Kombination auf ein Eingabepaar (Lesart) angewendet wird.
3. Zahl der Lesarten, die bei einem einzelnen Kombinationsschritt jeweils vorhanden sind.

### 11.3.2 Einfluß auf die Komplexität

- Faktor 1 ist grammatikunabhängig und wird als die Länge  $n$  in der Formel zu Charakterisierung der Komplexität verwendet.
- Faktor 2 ist eine grammatikabhängige Konstante.
- Nur Faktor 3 kann die Zahl der Regelanwendungen über das lineare Anwachsen mit der Länge der Eingabe hinaus in die Höhe treiben. Ob bei einer Eingabe mehrere Regeln eines Regelpakets erfolgreich sein können, hängt von den Eingabebedingungen der Regeln ab.

### 11.3.3 Mögliche Beziehungen zwischen Eingabebedingungen

1. *Inkompatible* Eingabebedingungen: Zwei Regeln haben inkompatible Eingabebedingungen, wenn es keine Eingabepaare gibt, die von beiden Regeln akzeptiert werden.
2. *Kompatible* Eingabebedingungen: Zwei Regeln haben kompatible Eingabebedingungen, wenn es mindestens ein Eingabepaar gibt, das von beiden Regeln akzeptiert wird, und mindestens ein Eingabepaar, das von der einen, aber nicht von der anderen Regel akzeptiert wird.
3. *Identische* Eingabebedingungen: Zwei Regeln haben identische Eingabebedingungen, wenn für alle Eingabepaare gilt, daß sie entweder von beiden Regeln akzeptiert oder von beiden Regeln abgelehnt werden.

### 11.3.4 Definition unambiger LA-Grammatiken

Eine LA-Grammatik ist unambig dann und nur dann, wenn (i) für alle Regelpakete gilt, daß deren Regeln *inkompatible* Eingabebedingungen haben, und (ii) es keine lexikalischen Ambiguitäten gibt.

### 11.3.5 Definition syntaktisch ambiger LA-Grammatiken

Eine LA-Grammatik ist syntaktisch ambig dann und nur dann, wenn (i) es mindestens ein Regelpaket gibt, das mindestens zwei Regeln mit *kompatiblen* Eingabebedingungen enthält, und (ii) es keine lexikalischen Ambiguitäten gibt.

### 11.3.6 Globale syntaktische Ambiguität

Eine syntaktische Ambiguität ist +global, wenn sie eine Eigenschaft des ganzen Satzes ist.

Beispiel: Flying air planes can be dangerous.

### 11.3.7 Lokale syntaktische Ambiguität

Eine syntaktische Ambiguität ist –global, wenn sie eine Eigenschaft von nur einem Teil des Satzes ist.

Beispiel: The horse raced by the barn fell.

### 11.3.8 Rolle der $\pm$ global Unterscheidung

In der LA-Grammatik besteht der Unterschied zwischen +globalen und –globalen Ambiguitäten allein darin, ob mehr als eine Lesart bis zum Satzende ‘überlebt’ oder nicht. Die  $\pm$ global Unterscheidung hat keinen Einfluß auf die Komplexität und wird hauptsächlich aus linguistischen Gründen gemacht.

### 11.3.9 Recursive syntaktische Ambiguität

Eine Ambiguität ist +rekursiv, wenn sie innerhalb einer rekursiven Regelschleife entsteht.

Beispiele: Die C-LAGs für  $WW^R$  (siehe 11.5.6) und  $WW$  (siehe 11.5.8), die –global sind, und für **SubsetSum** (siehe 11.5.11), die +global ist.

### 11.3.10 Nicht-rekursive syntaktische Ambiguität

Eine Ambiguität ist –rekursiv, wenn keiner der Ableitungszweige in den ambiguitätsverursachenden Zustand zurückführen kann.

Beispiele: Die C-LAGs für  $a^k b^k c^m d^m \cup a^k b^m c^m d^k$  (siehe 11.5.3), die +global ist, und die C-LAGs für natürliche Sprachen in den Kapiteln 17 und 18, die sowohl +global als auch –global ambig sind.

### 11.3.11 Rolle der $\pm$ rekursiv Unterscheidung

Die  $\pm$ recursive Unterscheidung ist entscheidend für die Komplexitätsanalyse, weil gezeigt werden kann, daß bei LA-Grammatiken mit nicht-rekursiven Ambiguitäten die maximale Anzahl der Regelanwendungen pro Kombination durch eine grammatikabhängige Konstante begrenzt ist.

### 11.3.12 Theorem 3

Wenn eine LA-Grammatik nur –rekursive Ambiguitäten enthält, dann ist die maximale Anzahl der Regelanwendungen

$$(n - (R - 2)) \cdot 2^{(R-2)}$$

für  $n > (R - 2)$ , wobei  $n$  die Länge der Eingabe und  $R$  die Anzahl der Regeln der LA-Grammatik ist.

*Beweis:* Das Parsen einer Eingabe der Länge  $n$  erfordert  $(n - 1)$  Kombinationsschritte. Wenn eine LA-Grammatik aus  $R$  Regeln besteht, dann muß spätestens nach  $R$  Kombinationen eine dieser Regeln erneut angewendet werden. Außerdem können in einer Kombination maximal  $R$  Regelanwendungen pro Lesart durchgeführt werden.

Gemäß der Definition –rekursiver Ambiguitäten dürfen Regeln, die eine syntaktische Ambiguität verursacht haben, in einem zeitlinearen Ableitungspfad (Lesart) nicht erneut angewendet werden. Bei ihrer ersten Anwendung kann eine Regel  $r_i$  also ein Maximum von  $R - 1$  neuen Pfaden erzeugen (unter der Annahme, daß das Regelpaket von  $r_i$  alle  $R$  Regeln der Grammatik außer  $r_i$  selbst enthält). Bei der Fortsetzung eines dieser Pfade kann die zweite Ambiguität erzeugende Regel  $r_j$  ein Maximum von  $R - 2$  neuen Pfaden erzeugen, usw. Wenn die verschiedenen Regeln der LA-Grammatik mit den jeweils noch möglichen maximalen Regelpaketen definiert werden, dann wird nach  $R - 2$  Kombinationsschritten ein Maximum von  $2^{(R-2)}$  Lesarten erreicht.

Q.E.D.

## 11.4 Komplexität von Grammatiken und Automaten

### 11.4.1 Wahl der primitiven Operation

The Griffith and Petrick data is not in terms of actual time, but in terms of “primitive operations.” They have expressed their algorithms as sets of nondeterministic rewriting rules for a Turing-machine-like device. Each application of one of these is a primitive operation. We have chosen as our primitive operation the act of adding a state to a state set (or attempting to add one which is already there). We feel that this is comparable to their primitive operation because both are in some sense the most complex operation performed by the algorithm whose complexity is independent of the size of the grammar and the input string.

[Griffith und Petrick machen ihre Komplexitätsangaben nicht in Form der tatsächlichen Zeitmessung, sondern auf der Grundlage “primitiver Operationen.” Sie formulierten ihre Algorithmen als Mengen nichtdeterministischer Ersetzungsregeln für ein Turing-Maschinen-artiges Gerät. Dabei ist jede Regelanwendung eine primitive Operation. Wir haben als unsere primitive Operation den Vorgang des Hinzufügens eines Zustands zur Zustandsmenge gewählt (bzw. den Versuch des Hinzufügens eines Zustands, der bereits vorhanden ist). Wir meinen, daß dies mit deren primitiver Operation vergleichbar ist: Beide sind gewissermaßen die komplexeste Operation der jeweiligen Algorithmen, die unabhängig von der Größe der Grammatik und der Länge der Eingabe ist.]

J. Earley 1970, p. 100

## 11.4.2 Primitive Operation der C-LAGs

Die primitive Operation einer C-LAG bzw. eines C-LAG-Parsers ist eine Regelanwendung (wobei auch ergebnislose Versuche gezählt werden).

## 11.5 Subhierarchie der C1-, C2- und C3-LAGs

### 11.5.1 Die Unterklasse der C1-LAGs

Eine C-LAG ist eine C1-LAG wenn sie –rekursiv ambig ist. Die Klasse der C1-LAGs parst in linearer Zeit und enthält alle deterministischen kontextfreien Sprachen, die von einem DPDA ohne  $\lambda$ -Bewegung erkannt werden, plus kontextfreie Sprachen mit –rekursiven Ambiguitäten, wie z. B.  $a^k b^k c^m d^m \cup a^k b^m c^m d^k$  sowie viele kontextsensitive Sprachen, z. B.  $a^k b^k c^k$ ,  $a^k b^k c^k d^k e^k$ ,  $\{a^k b^k c^k\}^*$ ,  $L_{square}$ ,  $L_{hast}^k$ ,  $a^{2^i}$ ,  $a^k b^m c^{k \cdot m}$  und  $a^{i!}$ , wobei letztere nicht einmal mehr eine Indexsprache ist.

### 11.5.2 C1-LAG für kontextsensitives $a^{2^i}$

$$LX =_{def} \{[a(a)]\}$$

$$ST_S =_{def} \{[(a) \{r_1\}]\}$$

$$r_1: (a) \quad (a) \Rightarrow (aa) \quad \{r_2\}$$

$$r_2: (aX) \quad (a) \Rightarrow (Xbb) \quad \{r_2, r_3\}$$

$$r_3: (bX) \quad (a) \Rightarrow (Xaa) \quad \{r_2, r_3\}$$

$$ST_F =_{def} \{[(aa) rp_1], [(bXb) rp_2], [(aXa) rp_3]\}.$$

### 11.5.3 C1-LAG für ambiges $a^k b^k c^m d^m \cup a^k b^m c^m d^k$

$LX =_{def} \{[a(a)], [b(b)], [c(c)], [d(d)]\}$

$ST_S =_{def} \{[(a) \{r_1, r_2, r_5\}]\}$

$r_1: (X) \quad (a) \Rightarrow (a X) \quad \{r_1, r_2, r_5\}$

$r_2: (a X) \quad (b) \Rightarrow (X) \quad \{r_2, r_3\}$

$r_3: (X) \quad (c) \Rightarrow (c X) \quad \{r_3, r_4\}$

$r_4: (c X) \quad (d) \Rightarrow (X) \quad \{r_4\}$

$r_5: (X) \quad (b) \Rightarrow (b X) \quad \{r_5, r_6\}$

$r_6: (b X) \quad (c) \Rightarrow (X) \quad \{r_6, r_7\}$

$r_7: (a X) \quad (d) \Rightarrow (X) \quad \{r_7\}$

$ST_F =_{def} \{[\varepsilon rp_4], [\varepsilon rp_7]\}$

### 11.5.4 Das Single Return Principle (SRP)

Eine rekursive Ambiguität ist *single return*, wenn genau einer der Fortsetzungszweige der Ambiguität in den Zustand zurückkehrt, der die Ambiguität verursachte.

### 11.5.5 Die Unterklasse der C2-LAGs

Eine C-LAG ist eine C2-LAG wenn sie (i) rekursive Ambiguitäten erzeugt und (ii) alle Ambiguitäten die Beschränkung des *Single-Return-Prinzips* erfüllen (SR-rekursive Ambiguität). Die Klasse der C2-Sprachen parst in polynomialer Zeit und enthält nondeterministische kontextfreie Sprachen wie  $WW^R$  und  $L_{hast}^\infty$ , plus kontextsensitive languages wie  $WW$ ,  $W^{k \geq 3}$ ,  $\{WWW\}^*$  und  $W_1W_2W_1^RW_2^R$ .

### 11.5.6 C2-LAG für kontextfreies $WW^R$

$$LX =_{def} \{[a(a)], [b(b)], [c(c)], [d(d)] \dots\}$$

$$ST_S =_{def} \{[(seg_c) \{r_1, r_2\}]\}, \text{ where } seg_c \in \{a, b, c, d, \dots\}$$

$$r_1: (X) (seg_c) \Rightarrow (seg_c X) \{r_1, r_2\}$$

$$r_2: (seg_c X) (seg_c) \Rightarrow (X) \{r_2\}$$

$$ST_F =_{def} \{[\varepsilon rp_2]\}$$

### 11.5.7 Die Ableitungsstruktur des *worst case* in $WW^R$

rules:

2

1 2 2

1 1 2 2 2

1 1 1 2 2

1 1 1 1 2

1 1 1 1 1

analyses:

a \$ a

a a \$ a a

a a a \$ a a a

a a a a \$ a a

a a a a a \$ a

a a a a a a \$

### 11.5.8 C2-LAG für kontextsensitives WW

$$LX =_{def} \{[a(a)], [b(b)], [c(c)], [d(d)] \dots\}$$

$$ST_S =_{def} \{[(seg_c) \{r_1, r_2\}]\}, \text{ where } seg_c \in \{a, b, c, d, \dots\}$$

$$r_1: (X) \quad (seg_c) \Rightarrow (X \text{ } seg_c) \{r_1, r_2\}$$

$$r_2: (seg_c X) \quad (seg_c) \Rightarrow (X) \quad \{r_2\}$$

$$ST_F =_{def} \{[\varepsilon \text{ } rp_2]\}$$

### 11.5.9 C2-LAG für kontextsensitives $W_1 W_2 W_1^R W_2^R$

$$LX =_{def} \{[a(a)], [b(b)]\}$$

$$ST_S =_{def} \{[(seg_c) \{r_{1a}\}], [(seg_c) \{r_{1b}\}]\}, \text{ where } seg_c, seg_d \in \{a, b\}$$

$$r_{1a}: (seg_c) \quad (seg_d) \Rightarrow (\# seg_c seg_d) \quad \{r_2, r_3\}$$

$$r_{1b}: (seg_c) \quad (seg_d) \Rightarrow (seg_d \# seg_c) \quad \{r_3, r_4\}$$

$$r_2: (X) \quad (seg_c) \Rightarrow (X seg_c) \quad \{r_2, r_3\}$$

$$r_3: (X) \quad (seg_c) \Rightarrow (seg_c X) \quad \{r_3, r_4\}$$

$$r_4: (X seg_c) \quad (seg_c) \Rightarrow (X) \quad \{r_4, r_5\}$$

$$r_5: (seg_c X \#) \quad (seg_c) \Rightarrow (X) \quad \{r_6\}$$

$$r_6: (seg_c X) \quad (seg_c) \Rightarrow (X) \quad \{r_6\}$$

$$ST_F =_{def} \{[\varepsilon rp_5], [\varepsilon rp_6]\}$$

### 11.5.10 The subclass of C3-LAGs

Eine C-LAG ist eine C3-LAG wenn sie unbeschränkte rekursive Ambiguitäten enthält. Die Unterklasse der C3-LAGs parst in exponentieller Zeit und enthält u.a. die deterministische kontextfreie Sprache  $L_{no}$ , die *hardest context-free language* HCFL, plus kontextsensitive Sprachen wie SubsetSum and SAT, die  $\mathcal{NP}$ -vollständig sind.

### 11.5.11 C3-LAG für SubsetSum.

$$LX =_{def} \{[0 (0)], [1 (1)], [\# (\#)]\}$$

$$ST_S =_{def} \{[(seg_c) \{r_1, r_2\}]\}, \text{ where } seg_c \in \{0, 1\}$$

$$seg_c \in \{0, 1\}$$

$$r_1: (X) \quad (seg_c) \quad \Rightarrow \quad (seg_c X) \{r_1, r_2\}$$

$$r_2: (X) \quad (\#) \quad \Rightarrow \quad (\# X) \quad \{r_3, r_4, r_6, r_7, r_{12}, r_{14}\}$$

$$r_3: (X seg_c) \quad (seg_c) \quad \Rightarrow \quad (0 X) \quad \{r_3, r_4, r_6, r_7\}$$

$$r_4: (X \#) \quad (\#) \quad \Rightarrow \quad (\# X) \quad \{r_3, r_4, r_6, r_7, r_{12}, r_{14}\}$$

$$r_5: (X seg_c) \quad (seg_c) \quad \Rightarrow \quad (0 X) \quad \{r_5, r_6, r_7, r_{11}\}$$

$$r_6: (X 1) \quad (0) \quad \Rightarrow \quad (1 X) \quad \{r_5, r_6, r_7, r_{11}\}$$

$$r_7: (X 0) \quad (1) \quad \Rightarrow \quad (1 X) \quad \{r_8, r_9, r_{10}\}$$

$$r_8: (X seg_c) \quad (seg_c) \quad \Rightarrow \quad (1 X) \quad \{r_8, r_9, r_{10}\}$$

$$r_9: (X 1) \quad (0) \quad \Rightarrow \quad (0 X) \quad \{r_5, r_6, r_7, r_{11}\}$$

$$r_{10}: (X 0) \quad (1) \quad \Rightarrow \quad (0 X) \quad \{r_8, r_9, r_{10}\}$$

$$r_{11}: (X \#) \quad (\#) \quad \Rightarrow \quad (\# X) \quad \{r_3, r_4, r_6, r_7, r_{12}, r_{14}\}$$

$$r_{12}: (X 0) \quad (seg_c) \quad \Rightarrow \quad (0 X) \quad \{r_4, r_{12}, r_{14}\}$$

$$r_{13}: (X 0) \quad (seg_c) \quad \Rightarrow \quad (0 X) \quad \{r_{11}, r_{13}, r_{14}\}$$

$$r_{14}: (X 1) \quad (seg_c) \quad \Rightarrow \quad (1 X) \quad \{r_{11}, r_{13}, r_{14}\}$$

$$ST_F =_{def} \{[(X) rp_4]\}$$

### 11.5.12 Restriktionstypen der LA-Grammatik

0. Typ-A: keine Beschränkung
1. Typ-B: Die Länge der Zwischenausdrucks-kategorien ist über  $B \cdot n$  beschränkt, wobei  $B$  eine Konstante und  $n$  die Gesamtlänge der Eingabe ist (*R1.1*, Aufwand).
2. Typ-C3: Die Form der Kategoriemuster resultiert in einer konstanten Beschränkung  $C$  der kategorialen Operationen (*R1.2*, Aufwand).
3. Typ-C2: Wie LA-Typ-C3 und die Grammatik ist höchstens *single return* rekursiv ambig (*R2*, Anzahl).
4. Typ-C1: Wie LA-Typ-C3 und die Grammatik ist höchstens –rekursiv ambig (*R2*, Anzahl).

### 11.5.13 LA-grammatische Hierarchie der formalen Sprachen

Beschränkung	Typen der LAG	Sprachklassen	Komplexitätsgrad
LA-Typ-C1	C1-LAGs	C1-Sprachen	linear
LA-Typ-C2	C2-LAGs	C2-Sprachen	polynomial
LA-Typ-C3	C3-LAGs	C3-Sprachen	exponentiell
LA-Typ-B	B-LAGs	B-Sprachen	exponentiell
LA-Typ-A	A-LAGs	A-Sprachen	rekursiv