

Computerlinguistik I

Vorlesung im WiSe 2018/2019
(M-GSW-09)

Prof. Dr. Udo Hahn

Lehrstuhl für Computerlinguistik
Institut für Germanistische Sprachwissenschaft
Friedrich-Schiller-Universität Jena

<http://www.julielab.de>

Grundbegriffe zur Syntaxanalyse von CFGs

- Das **Wort-** bzw. **Erkennungsproblem** für eine kontextfreie Grammatik G :
Zeige für $G = (N, T, P, S)$ und $\omega \in T^*$, dass ω von G (nicht) erzeugt werden kann (d.h.: $\omega \in \mathcal{L}(G)$ oder $\omega \notin \mathcal{L}(G)$).
Ein Algorithmus, der dieses Problem löst, heißt **Erkennungsalgorithmus** (oder **Recognizer**).

Grundbegriffe zur Syntaxanalyse von CFGs

- Das **Analyseproblem** für eine kontextfreie Grammatik G :

Bestimme für $G = (N, T, P, S)$ und $\omega \in T^*$ entweder eine syntaktische Struktur von ω bezüglich G oder zeige, dass $\omega \notin \mathcal{L}(G)$.

Ein Algorithmus, der dieses Problem löst, heißt **Analysealgorithmus** (oder **Parser**) .

Die Bestimmung der syntaktischen Struktur heißt **Syntaxanalyse** bzw. **Parsing** .

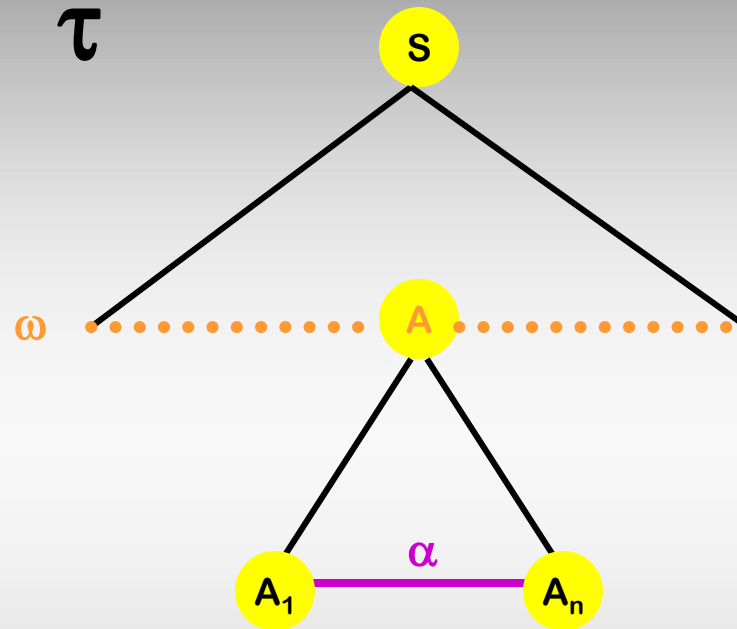
Bemerkungen zur Syntaxanalyse von CFGs

- Ein Analysealgorithmus löst mit der (fehlschlagenden) Bestimmung einer syntaktischen Struktur stets auch das Wortproblem.
- Für Typ-0-Grammatiken ist das Wortproblem unlösbar.
- Für Typ-1-Grammatiken, die bestimmten Beschränkungen unterliegen, und generell für Typ-2-Grammatiken ist das Wortproblem lösbar – wenn auch (für Typ-1) mit z.T. beträchtlicher, aber noch polynomialer Berechnungskomplexität.
- Für Typ-3-Grammatiken ist das Wort- und Analyseproblem einfach lösbar (linear).

Grundbegriffe zur Syntaxanalyse von CFGs

- Sei ω ein von der kontextfreien Grammatik G erzeugtes Wort und τ ein zugehöriger Strukturbaum, der eine feste (beliebig wählbare, aber dann gegebene) Verzweigung besitzt, die aus einem Knoten und seinen direkten Nachfolgern besteht. Diese Verzweigung beschreibt die Anwendung einer Produktion, etwa $A \rightarrow \alpha$ mit $\alpha = A_1 \dots A_n$ und $A_i \in \mathcal{V}$ für $1 \leq i \leq n$.

Gliederung eines Strukturbaums

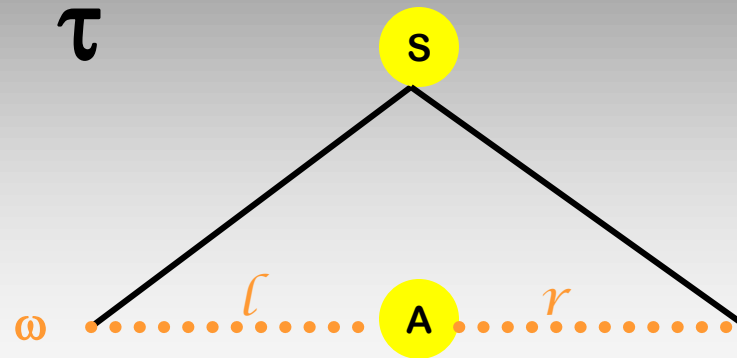


Grundbegriffe zur Syntaxanalyse von CFGs

- Durch die Fixierung einer festen Verzweigung und der zugehörigen Anwendung einer Produktion wird τ in **Teilstrukturen** zerlegt.

Dazu betrachten wir die Klasse τ_A aller Strukturbäume zu G , die Anfangsteilbäume von τ sind (d.h. die gleiche Wurzel besitzen) und den fest herausgegriffenen Knoten A als Endknoten haben. Diese haben Endschnittbilder der Form ℓAr mit $\ell, r \in \mathcal{V}^*$ und beschreiben die Ableitung: $S \stackrel{*}{\Rightarrow} \ell Ar$.

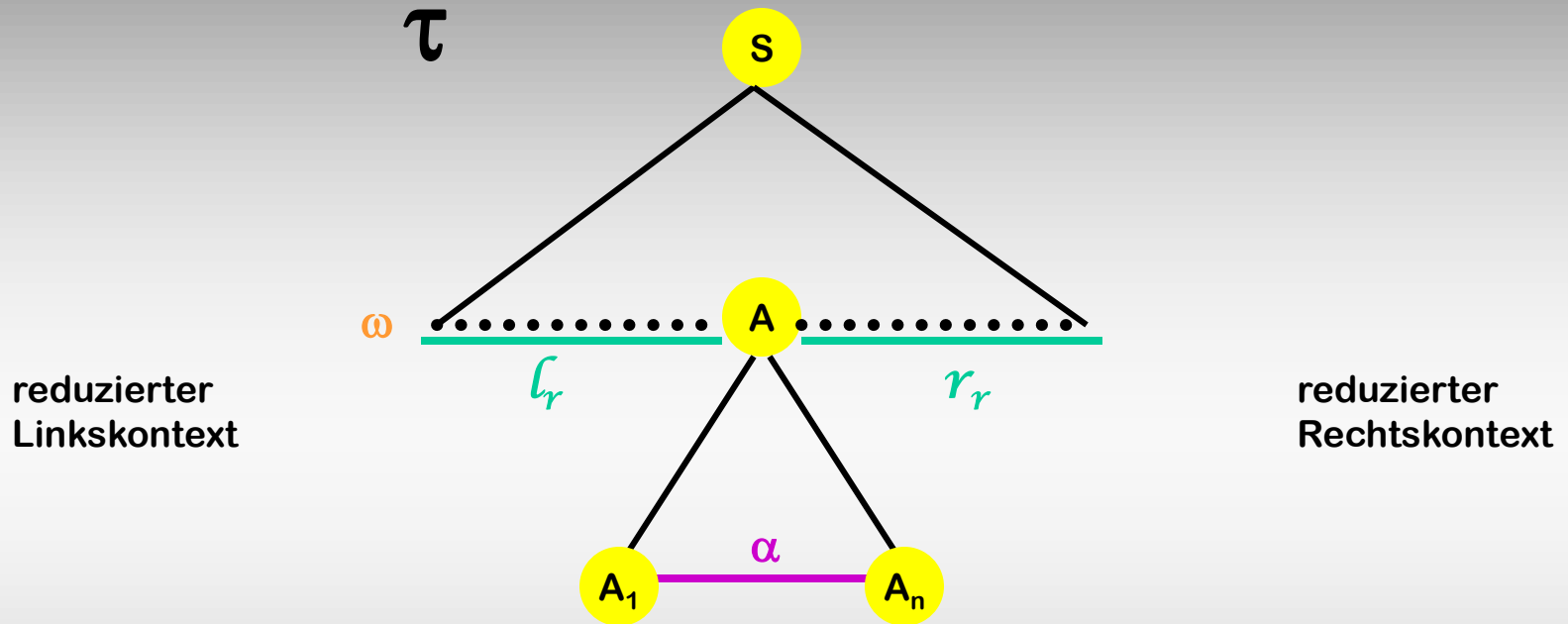
Gliederung eines Strukturbaums



Grundbegriffe zur Syntaxanalyse von CFGs

- Sei τ_{\min} der eindeutig fixierte Strukturbaum in τ_A mit minimaler Knotenzahl und $\ell_r A r_r$ sein Endschnittbild.
 ℓ_r ist der **reduzierte Linkskontext** und r_r der **reduzierte Rechtskontext** zur betrachteten Anwendung der Produktion $A \rightarrow \alpha$.

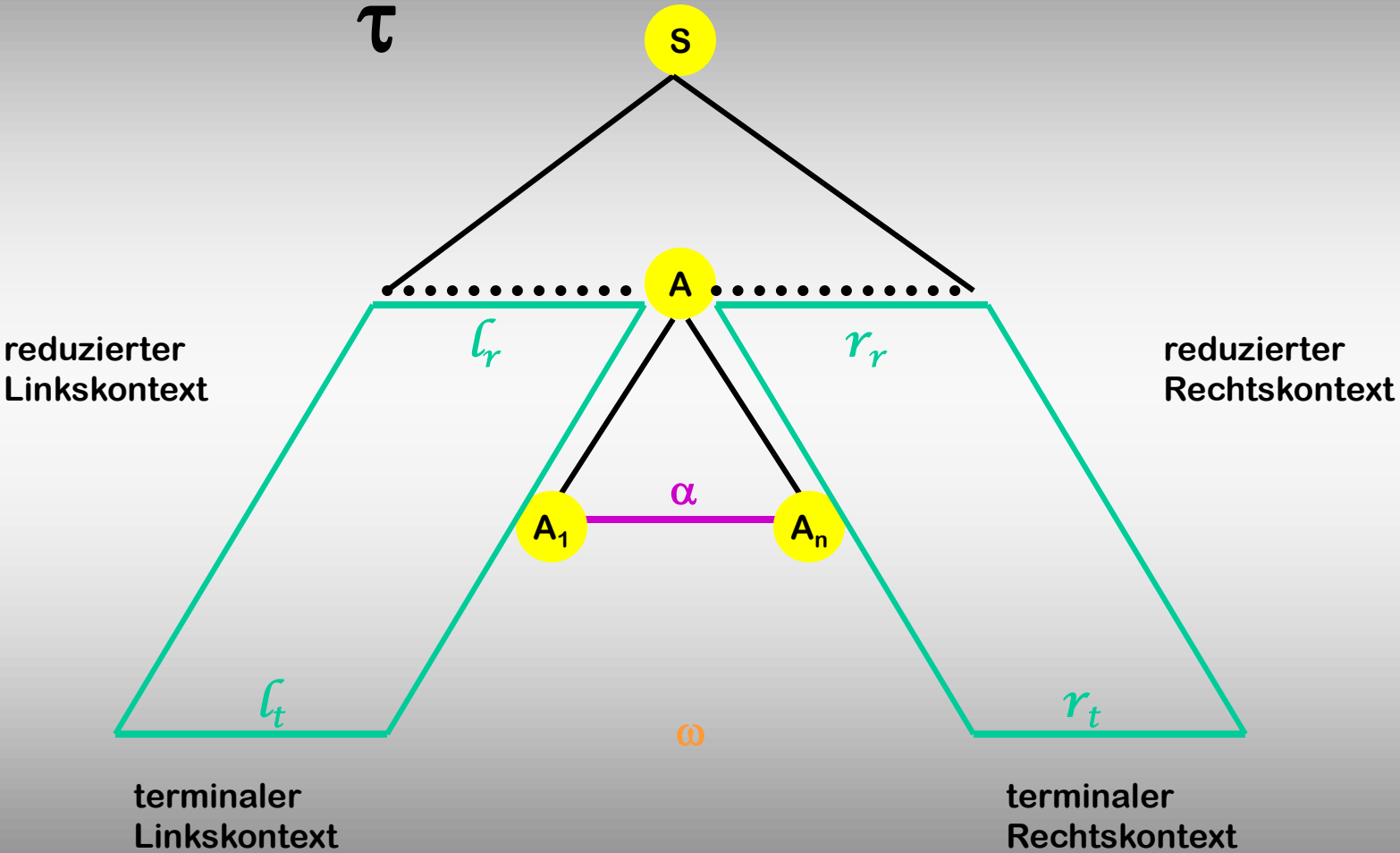
Gliederung eines Strukturbaums



Grundbegriffe zur Syntaxanalyse von CFGs

- Sei τ_{\max} der eindeutig fixierte Strukturbaum in τ_A mit maximaler Knotenzahl und $\ell_t A r_t$ sein Endschnittbild. Dann sind $\ell_t, r_t \in T^*$.
 ℓ_t heißt dann **terminaler Linkskontext** und r_t **terminaler Rechtskontext** zur betrachteten Anwendung der Produktion $A \rightarrow \alpha$.

Gliederung eines Strukturbaums

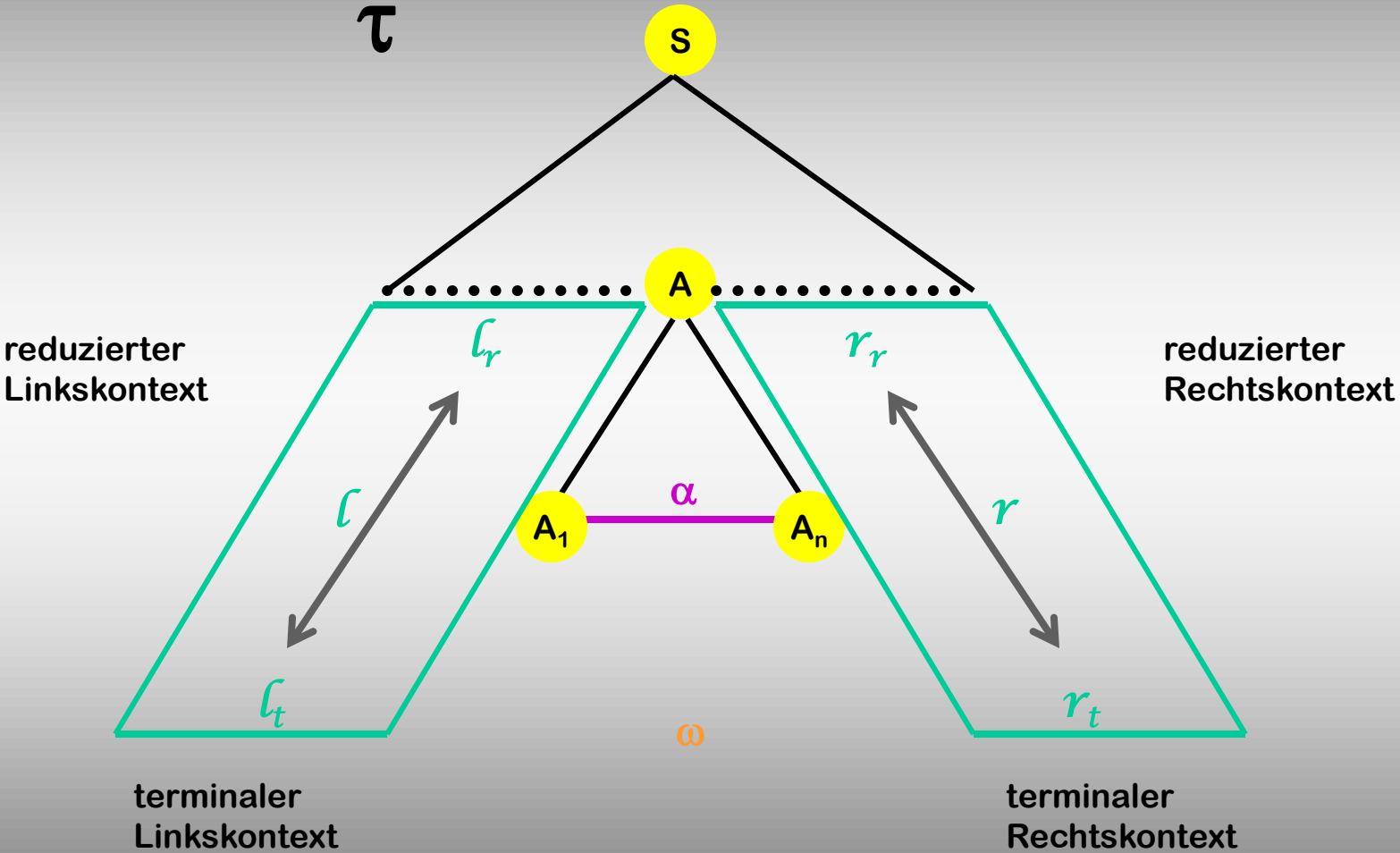


Grundbegriffe zur Syntaxanalyse von CFGs

- Sei τ_{bel} ein beliebig herausgegriffener Strukturbaum in τ_A und sei $\mathcal{L}_A r$ sein Endschnittbild. Dann gilt:

$$\mathcal{L}_r^* \Rightarrow \mathcal{L}^* \Rightarrow \mathcal{L}_t \quad \text{und} \quad r_r^* \Rightarrow r^* \Rightarrow r_t.$$

Gliederung eines Strukturbaums



Grundbegriffe zur Syntaxanalyse von CFGs

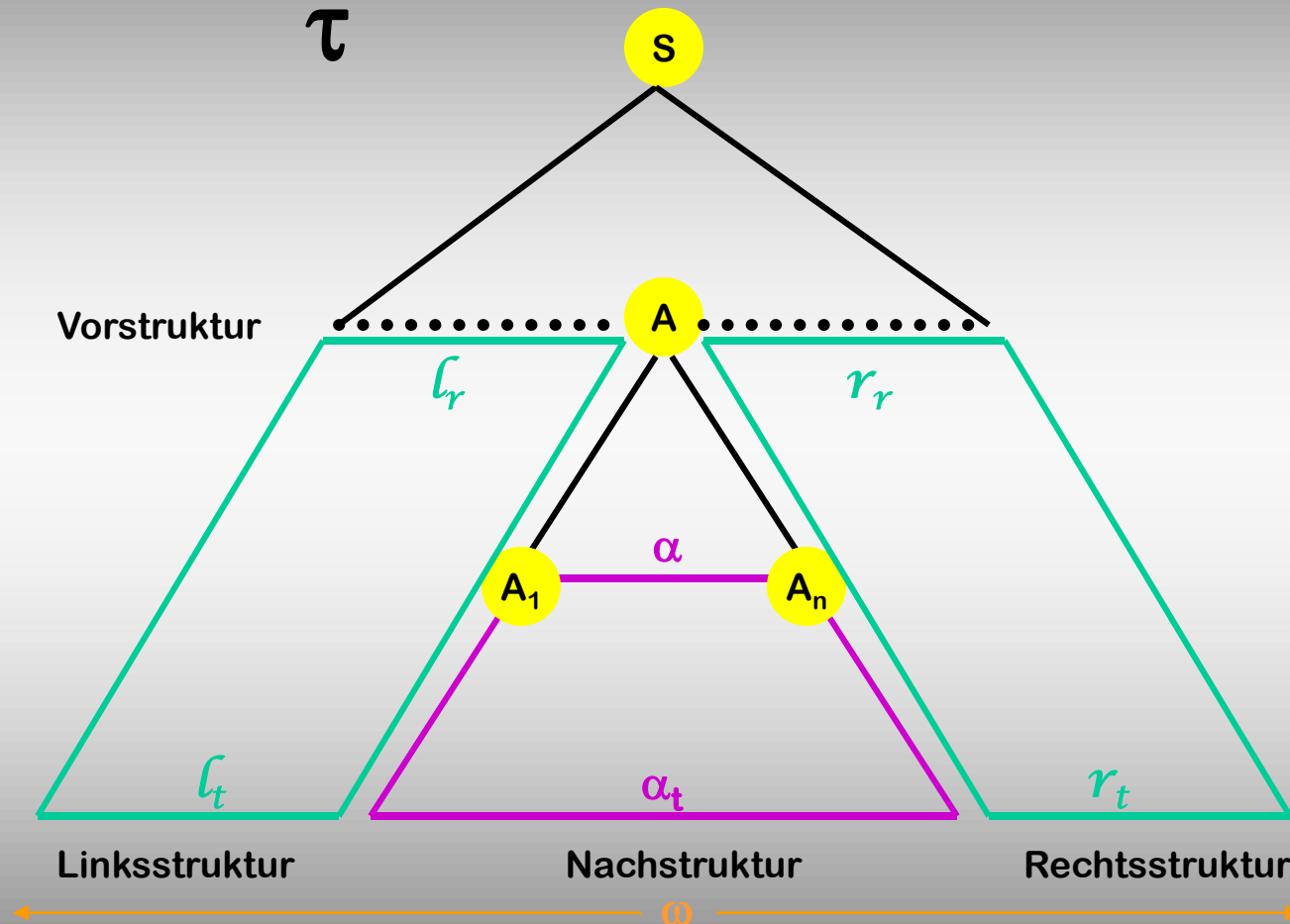
- Die fest herausgegriffene Anwendung der Produktion $A \rightarrow \alpha$ bestimmt in der betrachteten syntaktischen Struktur von ω somit vier Teilstrukturen, die Ableitungen für

$$S^* \Rightarrow \ell_r A r_r, \ell_r^* \Rightarrow \ell_t, \alpha^* \Rightarrow \alpha_t \text{ und } r_r^* \Rightarrow r_t$$

mit $\omega = \ell_t \alpha_t r_t$ entsprechen.

- Man nennt diese Teilstrukturen die zur herausgegriffenen Anwendung der Produktion $A \rightarrow \alpha$ gehörige **Vorstruktur**, **Linksstruktur**, **Nachstruktur** und **Rechtsstruktur**.

Gliederung eines Strukturbaums



Grundbegriffe zur Syntaxanalyse von CFGs

Die im Folgenden betrachteten **Analysestrategien** sind dadurch charakterisiert, dass bestimmte, zu einer festen Anwendung einer Produktion gehörige Teilstrukturen für das Erkennen der Anwendung dieser Produktion jeweils bekannt sein müssen.

- Erfolgt das Erkennen der Anwendung einer Produktion, sobald die zugehörige Linksstruktur total bekannt ist, spricht man von einer **Analyse von links nach rechts** (analog: **Analyse von rechts nach links**).

Grundbegriffe zur Syntaxanalyse von CFGs

In beiden Fällen ist die Reihenfolge des Auffindens der zugehörigen Vor- und Nachstruktur noch nicht festgelegt.

- Ist jeweils die Vorstruktur total bekannt ist, während die Nachstruktur noch unbekannt ist, spricht man von einer **abwärts gerichteten** (oder **Top-Down-**)**Analyse**. Im umgekehrten Fall – bei einer total erkannten Nachstruktur und noch unbekannter Vorstruktur – von einer **aufwärts gerichteten** (oder **Bottom-Up-**)**Analyse**.

Grundbegriffe zur Syntaxanalyse von CFGs

- Bei einer **Top-Down-Analyse** sind beim Erkennen der Anwendung einer Produktion $A \rightarrow \alpha$ die zugehörige Vorstruktur und Linksstruktur (und damit insbesondere auch der zugehörige terminale Linkskontext ℓ_t und der zugehörige reduzierte Rechtskontext r_r) bekannt.

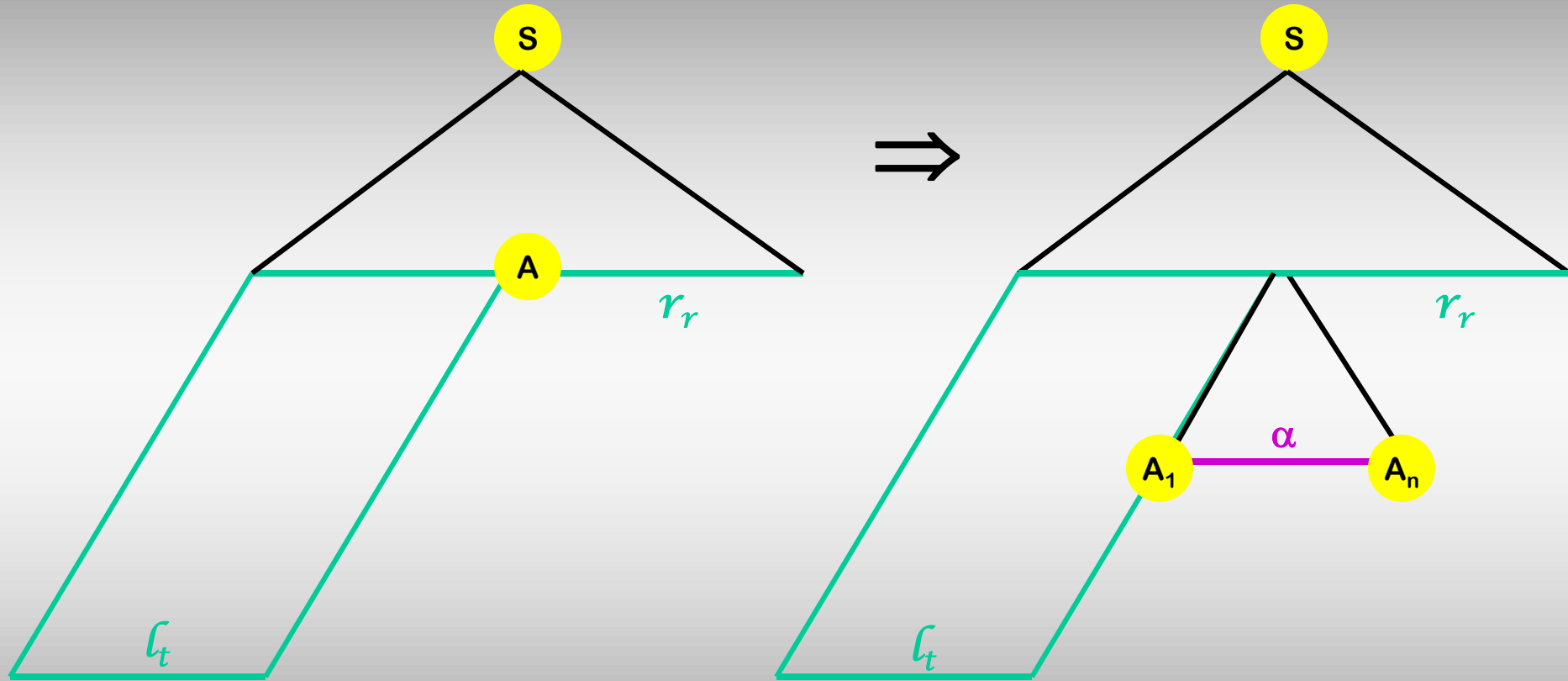
Grundbegriffe zur Syntaxanalyse von CFGs

- Dem Erkennen der Anwendung der Produktion $A \rightarrow \alpha$ entspricht der Übergang von der Satzform $s = \ell_t A r_r$ zur Satzform $s' = \ell_t \alpha r_r$, wobei s und s' die Endschnittbilder der jeweils erkannten Teile der Struktur sind.
- Dieser Übergang von s nach s' entspricht dem Ableitungsschritt $[\ell_t, A \rightarrow \alpha, r_r]$. Dabei liegt ℓ_t nach Voraussetzung immer in T^* .

Somit ergeben die bei der **Top-Down-Analyse** nacheinander gefundenen Ableitungsschritte eine **Linksableitung**, also:

$$\Delta = \{ \delta_i \}_{i=1}^n \text{ mit } \delta_i = [\ell_i, A_i \rightarrow \gamma_i, r_i], \ell_i \in T^* \text{ für } 1 \leq i \leq n .$$

Links-Rechts-Top-Down-Analyse



Grundbegriffe zur Syntaxanalyse von CFGs

- Bei einer **Bottom-Up-Analyse** sind beim Erkennen der Anwendung einer Produktion $A \rightarrow \alpha$ die zugehörige Nachstruktur und Linksstruktur (und damit insbesondere auch der zugehörige reduzierte Linkskontext ℓ_r , die rechte Seite der Produktion $A \rightarrow \alpha$ und der zugehörige terminale Rechtskontext r_t) bekannt.

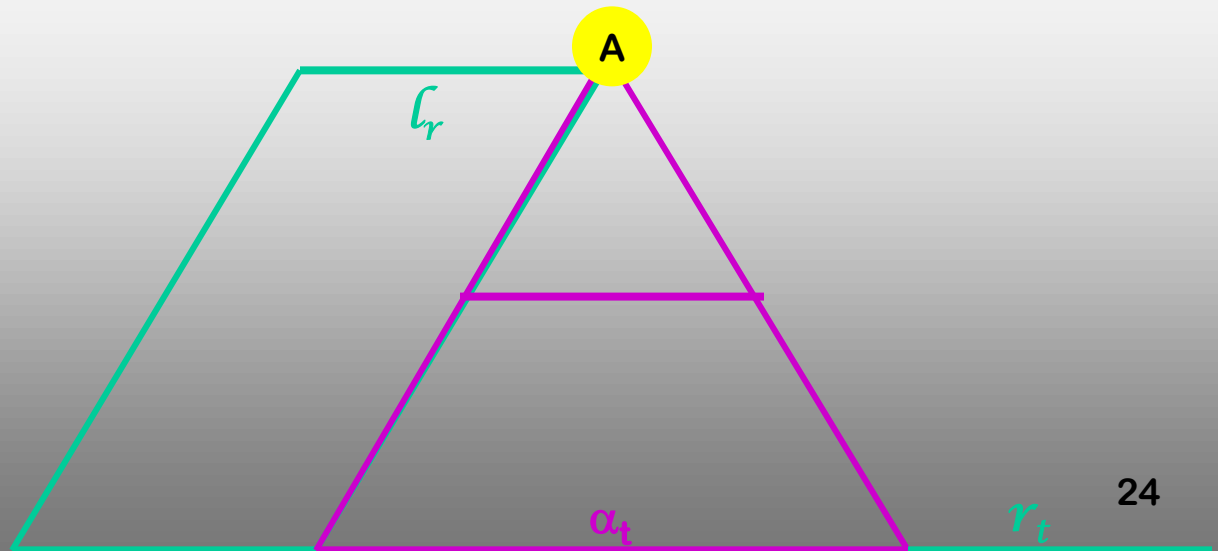
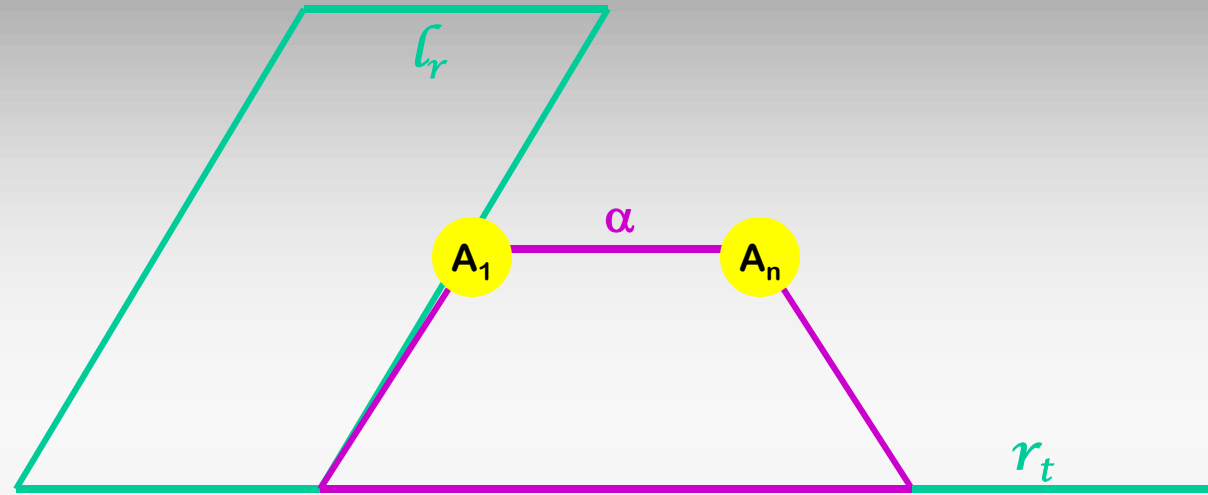
Grundbegriffe zur Syntaxanalyse von CFGs

- Der Anwendung der Produktion $A \rightarrow \alpha$ entspricht die Reduktion der Satzform $s = \ell_r \alpha r_t$ auf die Satzform $s' = \ell_r A r_t$, wobei s und s' als Endschnittbilder der jeweils noch unbekanntesten Teile zugleich Anfangsschnittbilder der jeweils erkannten Teile der Struktur sind.
- Dieser Übergang von s nach s' entspricht dem Reduktionsschritt $[\ell_r, A \rightarrow \alpha, r_t]$. Dabei liegt r_t nach Voraussetzung immer in T^* .

Somit ergeben die bei der **Bottom-Up-Analyse** nacheinander gefundenen Ableitungsschritte eine **Rechtsreduktion**, also:

$$\Delta = \{ \delta_i \}_{i=1}^n \text{ mit } \delta_i = [\ell_i, A_i \rightarrow \gamma_i, r_i], r_i \in T^* \text{ f\"ur } 1 \leq i \leq n$$

Links-Rechts-BottomUp-Analyse



Beispielgrammatik (CFG)

Syntax

S \rightarrow NP VP

NP \rightarrow Det N

NP \rightarrow NP Conj NP¹

VP \rightarrow V NP

Det \rightarrow ε ²

Lexikon

Det \rightarrow the

Conj \rightarrow and

N \rightarrow dog

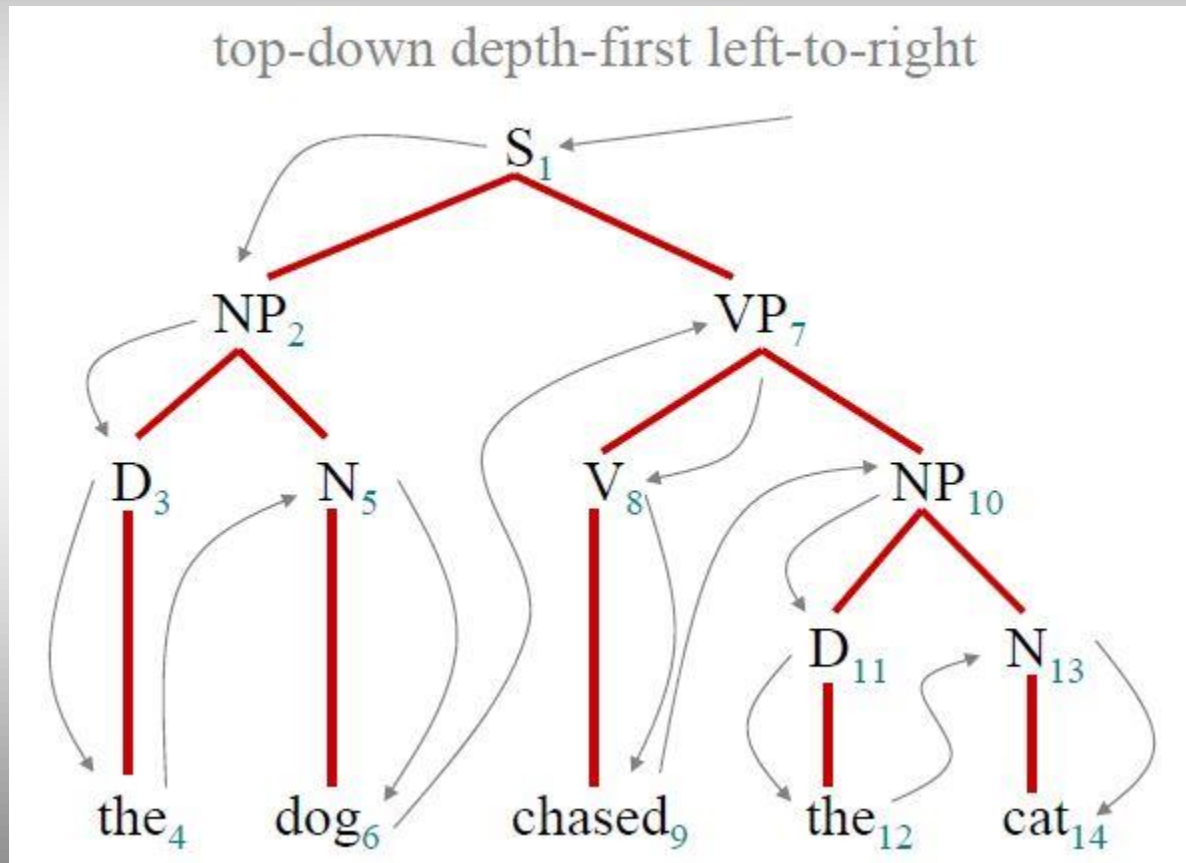
N \rightarrow cat

V \rightarrow chases

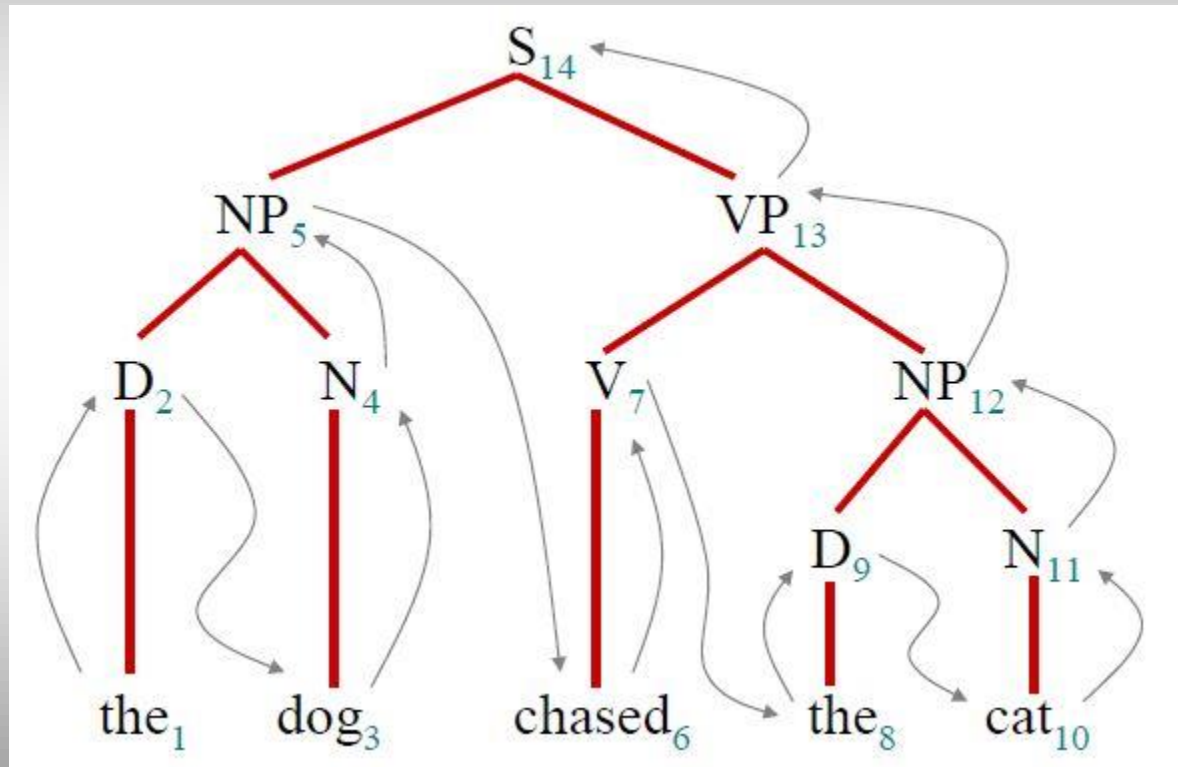
¹ nicht für top-down-Parser

² nicht für bottom-up-Parser

Top-Down-Traversierung



Bottom-Up-Traversierung



Shift-Reduce-Parsing

Der Shift-Reduce-Algorithmus verfolgt eine Bottom-Up-Strategie.

1. **Shift**: lege ein Wort aus der Eingabekette auf einen Stapel.
2. **Reduce**: reduziere den Stapel mit Hilfe der Grammatik soweit wie möglich.
3. Falls die Eingabekette noch Wörter enthält, gehe zu Shift, sonst halte.

Shift-Reduce-Parsing- Beispiel

Schritt	Aktion	Stack	Input String
	Start		the dog barked
1	Shift	the	dog barked
2	Reduce	D	dog barked
3	Shift	D dog	barked
4	Reduce	D N	barked
5	Reduce	NP	barked
6	Shift	NP barked	
7	Reduce	NP V	
8	Reduce	NP VP	
9	Reduce	S	

Grundbegriffe zur Syntaxanalyse von CFGs

- Sind die gerade betrachteten Übergänge der Satzform $s = \ell_t A r_r$ auf $s' = \ell_t \alpha r_r$ bei der Top-Down-Analyse bzw. $s = \ell_r \alpha r_t$ auf $s' = \ell_r A r_t$ bei der Bottom-Up-Analyse für $\ell_t, r_t \in T^*$ nicht erfüllt (d.h., es liegt keine Linksableitung bzw. Rechtsreduktion vor), ist die Ableitung bzw. Reduktion in Bezug auf ihre Analysestrategie nicht mehr eindeutig determiniert (d.h., es existieren mehrere Kontrollwörter / Parses für eine Ableitung);
sie heißt dann **nicht-deterministisch**.