4 Syntax

Inhalt

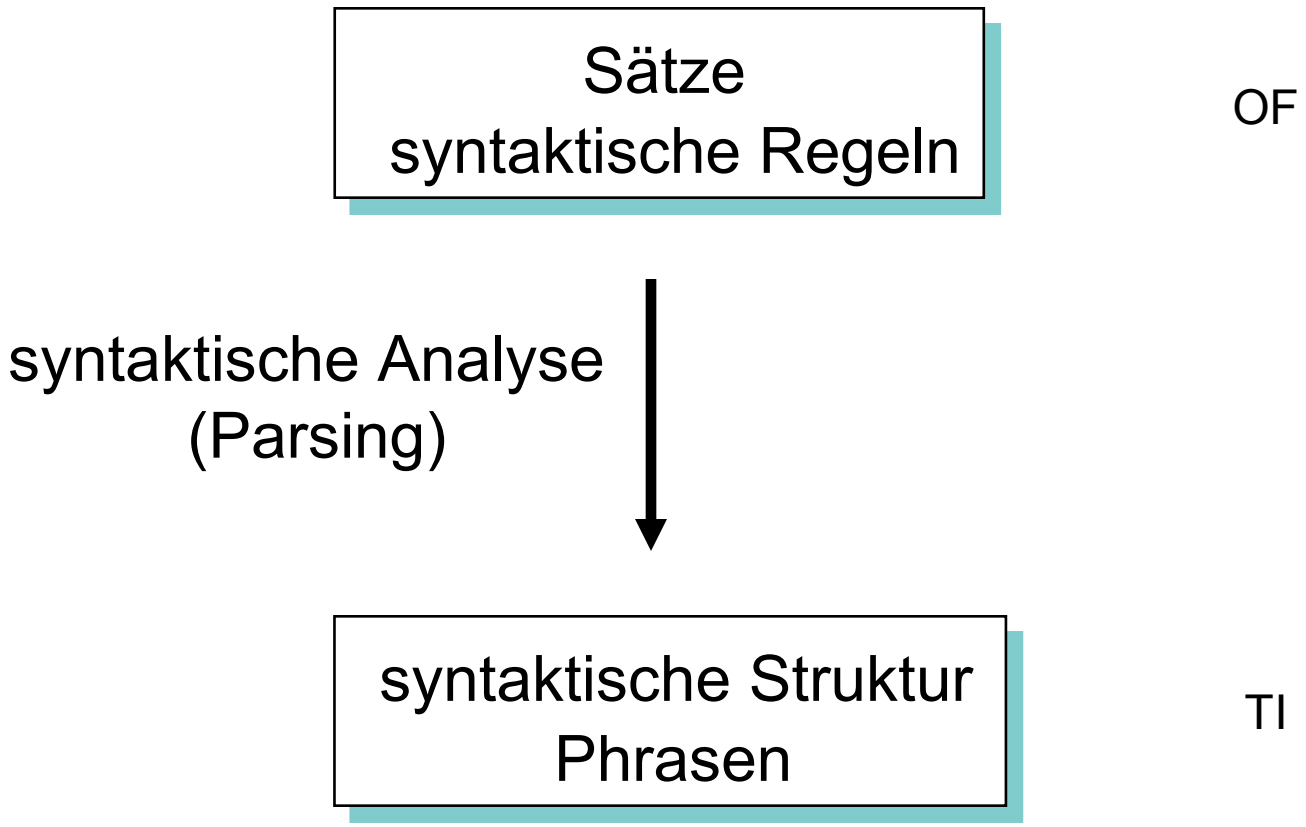
- Einführung
- Formale Sprachen und Grammatiken
- Übergangsnetze
- Merkmalsstrukturen – Unifikationsgrammatiken

4.1 Einführung

Einführung

- Oberflächenstruktur (OF)
  - äußere Erscheinungsform eines Satzes
- Tiefenstruktur (TI)
  - die Tiefenstruktur eines Satzes ist seine Repräsentation auf den nachfolgenden Ebenen
    - syntaktische Struktur
    - semantische Interpretation
    - oder etwas dazwischen

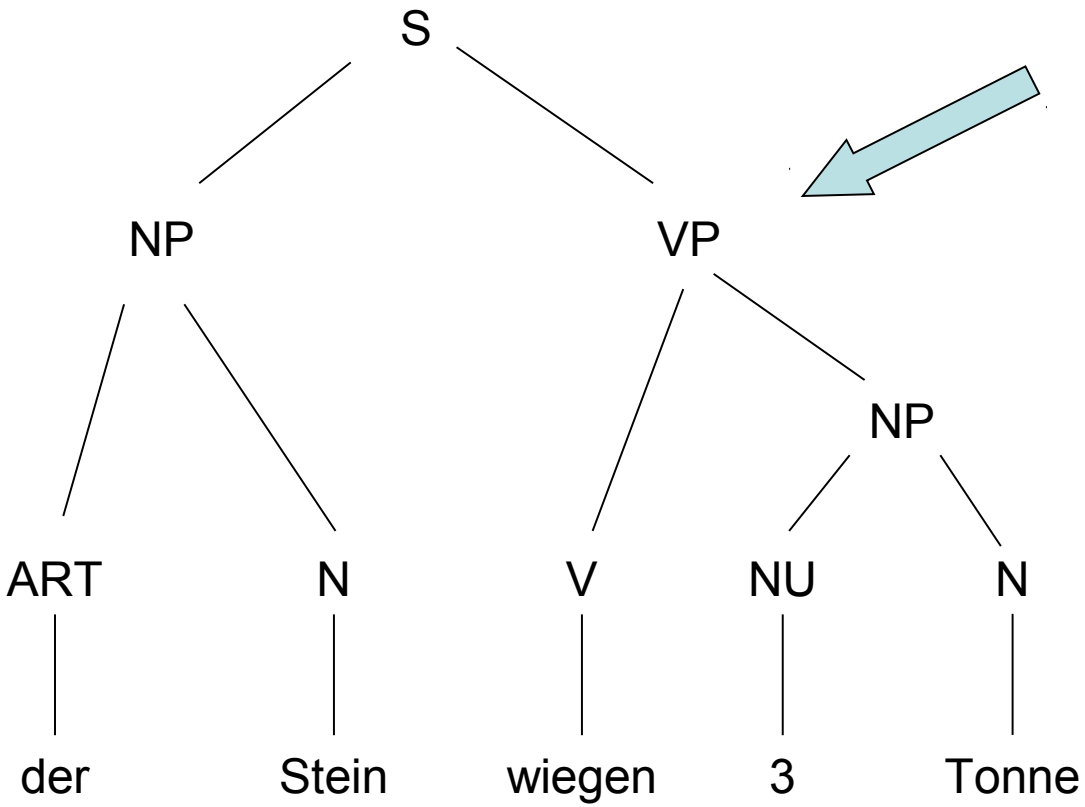
Syntax


Sätze
syntaktische Regeln
OF
syntaktische Analyse
(Parsing)
syntaktische Struktur
Phrasen
TI

Beispiel

Der Stein wiegt 3 Tonnen. ← **Oberflächenstruktur**

Tiefenstruktur



S – Satz
NP – Nominalphrase
VP – Verbalphrase
ART – Artikel
N – Substantiv (Nomen)
V – Verb
NU – Zahl

Grammatik – Phrasenstrukturgrammatik (PSG)

- Regelwerk, das es gestattet, die Sätze einer Sprache von den nicht zu dieser Sprache gehörenden Sätzen zu unterscheiden
- analytische Grammatik
  - jeder Oberflächenstruktur eines akzeptierten Satzes eine entsprechende Tiefenstruktur zuzuordnen
- generative Grammatik
  - aus vorgegebenen Tiefenstrukturen entsprechende Oberflächenstrukturen abzuleiten

Methoden zur Syntaxanalyse

- Grammatiken (Phrasenstrukturgrammatiken)
- Übergangsnetze (ATN)
- Merkmalsstrukturen - Unifikationsgrammatiken

4.2 Formale Sprachen und Grammatiken

4.2.1 Definitionen

Grammatik

$$G = (V_N, V_T, R, S)$$

V	endliche Menge von Symbolen (Alphabet)		
V_N	nichtterminale Symbole	$V_N \subseteq V$	$V_N \cup V_T = V$
V_T	terminale Symbole	$V_T \subseteq V$	$V_N \cap V_T = \emptyset$
S	Startsymbol	$S \in V_N$	

V^*	Menge aller Symbolketten über V		$\lvert\alpha\rvert, \alpha \in V^*$
L	Sprache	$L \subseteq V^*$	Länge einer Symbolkette
ε	leere Symbolkette		$\alpha^n, \alpha \in V^*, n \in N$
			α n - mal hintereinander

Grammatik

$$G = \left(V_N, V_T, R, S\right)$$

R endliche Menge von Regeln

$$R \subseteq \left(V^* \setminus V_T^*\right) \times V^*$$

$$r \in R \qquad r = \left(\varphi, \psi\right) \qquad r : \varphi \rightarrow \psi \qquad \varphi \neq \varepsilon$$

$$\exists\, r = \left(S, \psi\right) \in R$$

Beispiel

$$G_1 = \left(\{S, A, B\}, \{a, b\}, R, S\right)$$

Regeln:

$$S \rightarrow A$$

$$S \rightarrow B$$

$$A \rightarrow aa$$

$$A \rightarrow aSa$$

$$B \rightarrow b$$

$$B \rightarrow bBb$$

Ableitung

$$G = \left(V_N, V_T, R, S\right) \qquad \alpha, \beta \in V^*$$

1. *β ist aus α* **direkt ableitbar** *($\alpha \Rightarrow \beta$) genau dann, wenn*

$$\alpha = c_1 \varphi c_2, \quad \beta = c_1 \psi c_2$$

wobei

$$r : \varphi \rightarrow \psi, \quad r \in R, \quad c_1, c_2 \in V^*.$$

2. *β ist aus α* **ableitbar** *($\alpha \overset{*}{\Rightarrow} \beta$) genau dann, wenn existiert eine Folge*

$$A = \alpha_1, \alpha_2, \cdots, \alpha_n \quad (n \geq 1; \alpha_i \in V^*)$$

wobei

$$\alpha = \alpha_1, \quad \beta = \alpha_n, \quad \alpha_i \Rightarrow \alpha_{i+1} \quad (1 \leq i \leq n-1).$$

A wird als **Ableitung** *von β aus α bezeichnet.*

Beispiel

Aus der Grammatik G_1 läßt sich z.B. die Symbolkette „$aabbbaa$ “ableiten.

hergeleitete Form	Art der Herleitung
S	(Startsymbol)
A	(Anwendung der Regel) $S \rightarrow A$
aSa	$A \rightarrow aSa$
aAa	$S \rightarrow A$
$aaSaa$	$A \rightarrow aSa$
$aaBaa$	$S \rightarrow B$
$aabBbaa$	$B \rightarrow bBb$
$aabbbaa$	$B \rightarrow b$

Sprache

$$G = \left(V_N, V_T, R, S\right)$$

durch G erzeugte Sprache

$$L_G = L(G) = \left\{ \varphi \in V_T^* : S \overset{*}{\Rightarrow} \varphi \right\}$$

G_1 (schwach) äquivalent G_2

$\Leftrightarrow$ $L(G_1) = L(G_2)$

Grundtypen von Grammatiken
Chomsky Hierarchie

$$G = (V_N, V_T, R, S)$$

Typ 0 (uneingeschränkte Grammatik):

keine weiteren Einschränkungen an die Regelmenge *R*

Typ 1 (kontextsensitive Grammatik):

$r \in R \qquad r : \varphi \rightarrow \psi \qquad |\varphi| \leq |\psi|$

Die Anwendung einer Regel kann niemals zu einer Verkürzung der abgeleiteten Zeichenkette führen.

$r : c_1 A c_2 \rightarrow c_1 \omega c_2 \qquad A \in V_N \qquad \omega \neq \varepsilon \qquad c_1, c_2, \omega \in V^*$

Kontextfreie Grammatiken

$$G = \left(V_N, V_T, R, S\right)$$

Typ 2 (kontextfreie Grammatik):

$r : A \rightarrow \omega$

$A \in V_N \quad \omega \in V^* \quad (\omega \neq \varepsilon)$

Reguläre Grammatiken

Eine Grammatik $G = (V_N, V_T, R, S)$ *heißt* **reguläre Grammatik** *oder* **Grammatik vom Typ 3**, *wenn jede Regel* $r \in R$ *die Form*

$$r : A \to aB \quad oder \quad r : A \to a \quad , wobei \quad A, B \in V_N \quad ; \quad a \in V_T^*$$

hat.

Bemerkungen:

- es genügt schon: $a \in V_T$ oder $a = \varepsilon$
- Grammatiken der obigen Definition heißen **rechtslinear**
- **linkslinear** bedeutet: $r : A \to Ba$ oder $r : A \to a$, wobei $A, B \in V_N, a \in V_T^*$
- rechtslinear kann durch linkslinear ersetzt werden

Anwendungsmöglichkeiten

- Natürlichsprachliche Sätze (Syntax)
  - Terminalsymbole → Wortformen
  - Startsymbol → Satz
- Wortformen (Morphologie)
  - Terminalsymbole → Morpheme
  - Startsymbol → Wortform

Beispiele

$L_3 = \{ (ab)^n : n \in N \}$ Typ 3 Sprache

$L_2 = \{ a^n b^n : n \in N \}$ Typ 2 Sprache, aber nicht Typ 3

$L_1 = \{ a^n b^n c^n : n \in N \}$ Typ 1 Sprache, aber nicht Typ 2

$L_0 = \{ a^{2^n} : n \in N \}$ Typ 0 Sprache, aber nicht Typ 1

Bemerkungen

- Die Typ-0 und die kontextabhängigen Sprachen sind sehr allgemein gehalten. Sie finden in der Sprachverarbeitung wenig Anwendung.
- Reguläre Sprachen sind gut erforscht. Allerdings sind sie nur selten zur Abbildung etwa des Deutschen geeignet. Selten deshalb, weil sie für einige einfache Anwendungen wie beispielsweise zu einer natürlichsprachlichen Dialogschnittstelle eines Betriebssystems durchaus geeignet sind.
- Für die meisten Implementationen automatischer Sprachverarbeitung wurden kontextfreie Sprachen gewählt. Obwohl diese Sprachen deutlichen Einschränkungen unterliegen, können doch große Teile einer natürlichen Sprache kontextfrei behandelt werden. Auch sind für dieses Konzept effiziente Methoden der Sprachanalyse bekannt.

Präterminale Symbole

präterminal: $P \in V_N$

$\exists r : P \rightarrow a, \text{ wobei } a \in V_T^*$

Oft werden die präterminalen Symbole selbst wie die terminalen Symbole der Grammatik behandelt, um nicht für jedes Wort eine eigene Regel zu schreiben. Die Zuordnung der Wörter wird außerhalb der eigentlichen Grammatik durch Lexikonzugriff erledigt. Dies führt zu folgender Definition.

Phrasenstrukturgrammatik mit Lexikon

$$G_L = (V_N, V_P, L, R, S)$$

V	endliche Menge von Symbolen (Alphabet)		
V_N	nichtterminale Symbole	$V_N \subseteq V$	$V_N \cup V_P = V$
V_P	präterminale Symbole	$V_P \subseteq V$	$V_N \cap V_P = \emptyset$
S	Startsymbol	$S \in V_N$	
V^*	Menge aller Symbolketten über V		
L	Sprache	$L \subseteq V^*$	
ε	leere Symbolkette		

Phrasenstrukturgrammatik mit Lexikon

$$G_L = (V_N, V_P, L, R, S)$$

R endliche Menge von Regeln

$$R \subseteq (V^* \setminus V_P^*) \times V^*$$

$$r \in R \qquad r = (\varphi, \psi) \qquad r : \varphi \rightarrow \psi \qquad \varphi \neq \varepsilon$$

$$\exists\, r = (S, \psi) \in R$$

Lexikon: $L : \{\text{Wörter}\} \rightarrow V_P$

4.2.2 Kontextfreie Grammatiken und natürliche Sprache

Beispiel 1

$V_N = \{S, NP, VP, N, V, ART\}$

$V_T = \{\text{Hund}, \text{schläft}, \text{der}\}$

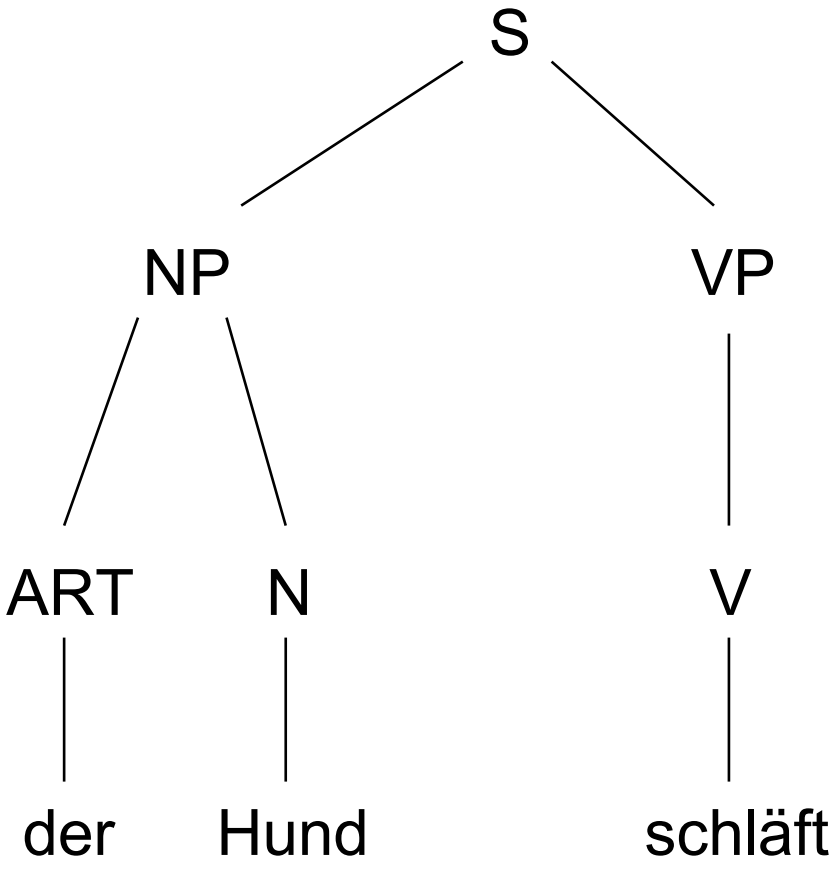
Regeln:

S	→	NP VP
NP	→	N
NP	→	ART N
VP	→	V
N	→	Hund
V	→	schläft
ART	→	der

Bezeichnungen

- ***S*** – Satz
- ***NP*** – Nominalphrase
- ***VP*** – Verbalphrase
- ***N*** – Nomen (Substantiv)
- ***V*** – Verb
- ***ART*** – Artikel
- ***ADJ*** – Adjektiv
- ***P*** – Präposition
- ***PP*** – Präpositionalphrase

Beispiel 1



S → NP VP

NP → N

NP → ART N

VP → V

N → Hund

V → schläft

ART → der

Beispiel 2

$$V_N = \{S, NP, VP, N, V, ART, ADJ\}$$

$$V_T = \left\{ \begin{array}{l} \text{Günter, Wasser, Hund,..., scheut, ärgert, ist,...,} \\ \text{blond, gut, gute,..., der, die, das,...} \end{array} \right\}$$

Beispiel 2 – Regeln

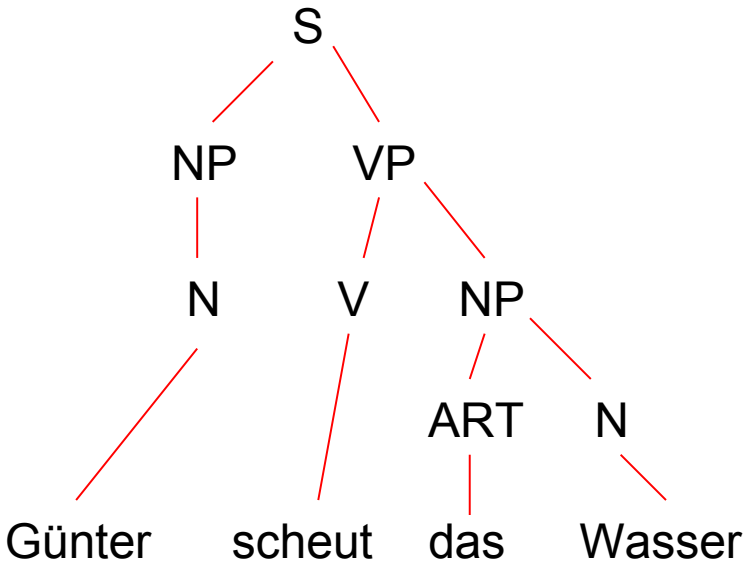
$$
\begin{aligned}
S &\rightarrow NP\ \ VP \\
NP &\rightarrow N \\
NP &\rightarrow ART\ \ N \\
NP &\rightarrow ADJ\ \ N \\
NP &\rightarrow ART\ \ ADJ\ \ N \\
VP &\rightarrow V \\
VP &\rightarrow V\ \ NP \\
VP &\rightarrow V\ \ ADJ
\end{aligned}
$$

$$
\begin{aligned}
N &\rightarrow Günter \\
N &\rightarrow Wasser \\
N &\rightarrow Hund \\
&\dots \\
V &\rightarrow scheut \\
V &\rightarrow aergert \\
V &\rightarrow ist \\
&\dots \\
ADJ &\rightarrow blond \\
ADJ &\rightarrow gut \\
ADJ &\rightarrow gute \\
&\dots \\
ART &\rightarrow der \\
ART &\rightarrow die \\
ART &\rightarrow das \\
&\dots
\end{aligned}
$$

Symbolketten (Sätze), die abgeleitet werden können

Günter scheut das Wasser.

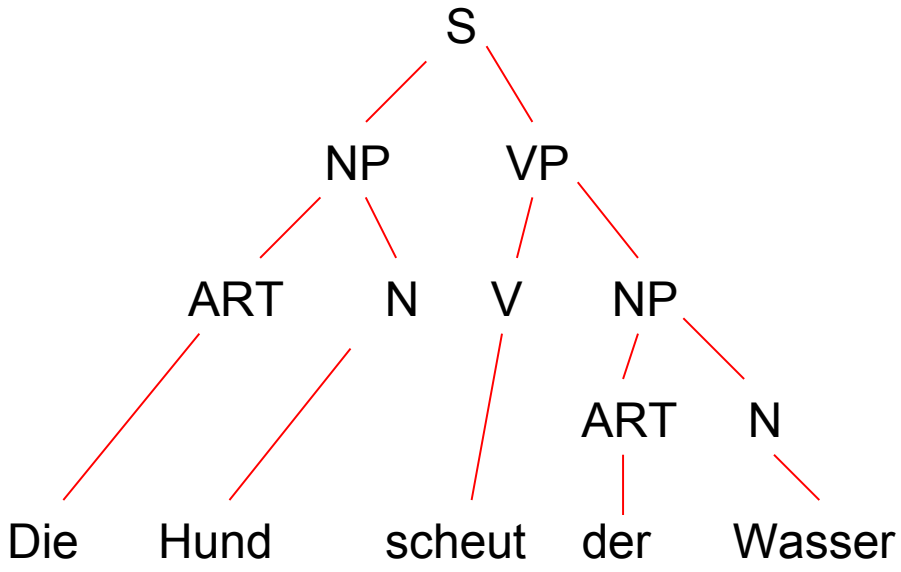
$$
\begin{aligned}
S &\rightarrow NP \quad VP \\
NP &\rightarrow N \\
NP &\rightarrow ART \quad N \\
NP &\rightarrow ADJ \quad N \\
NP &\rightarrow ART \quad ADJ \quad N \\
VP &\rightarrow V \\
VP &\rightarrow V \quad NP \\
VP &\rightarrow V \quad ADJ
\end{aligned}
$$



Symbolketten (Sätze), die abgeleitet werden können

Die Hund scheut der Wasser.

$$
\begin{aligned}
S &\rightarrow NP \quad VP \\
NP &\rightarrow N \\
NP &\rightarrow ART \quad N \\
NP &\rightarrow ADJ \quad N \\
NP &\rightarrow ART \quad ADJ \quad N \\
VP &\rightarrow V \\
VP &\rightarrow V \quad NP \\
VP &\rightarrow V \quad ADJ
\end{aligned}
$$



Präterminale Symbole

$V_P = \{N, V, ART, ADJ\}$

$$
\begin{aligned}
S &\rightarrow NP \;\; VP \\
NP &\rightarrow N \\
NP &\rightarrow ART \;\; N \\
NP &\rightarrow ADJ \;\; N \\
NP &\rightarrow ART \;\; ADJ \;\; N \\
VP &\rightarrow V \\
VP &\rightarrow V \;\; NP \\
VP &\rightarrow V \;\; ADJ
\end{aligned}
$$

Lexikon (L):

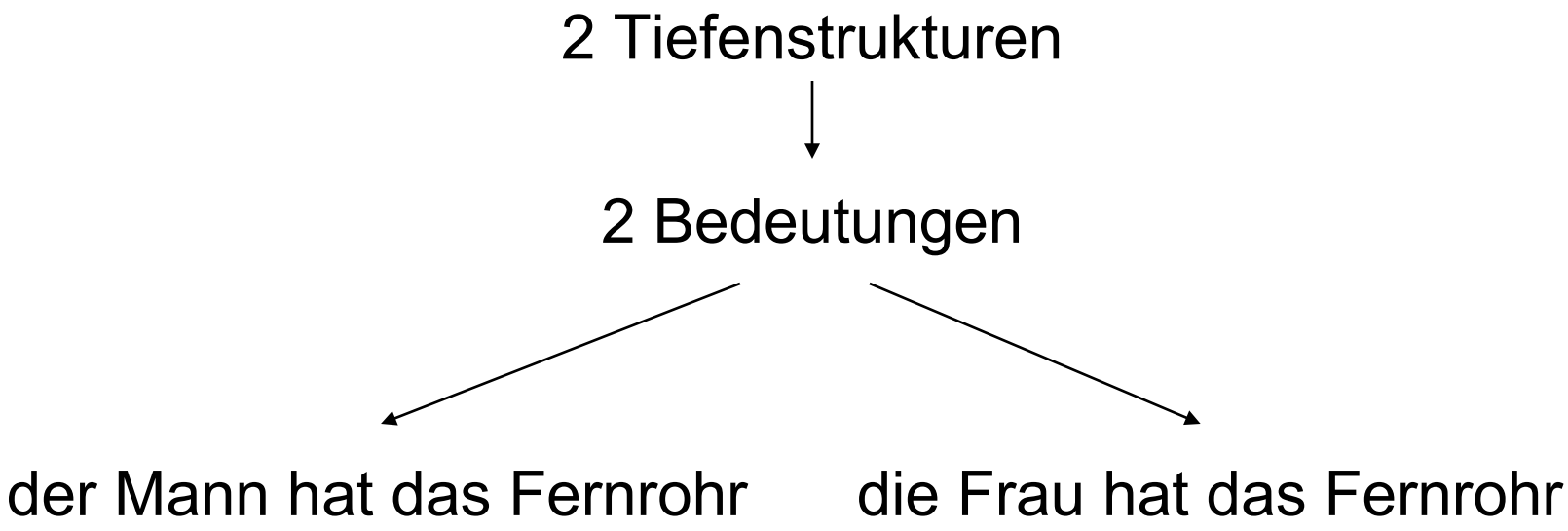
Günter	→	N
ist	→	V
gut	→	ADJ
der	→	ART
…		

Beispiel 3 – weitere Regeln

$$
\begin{aligned}
NP &\rightarrow NP \quad PP \\
VP &\rightarrow V \quad NP \quad PP \\
PP &\rightarrow P \quad NP
\end{aligned}
$$

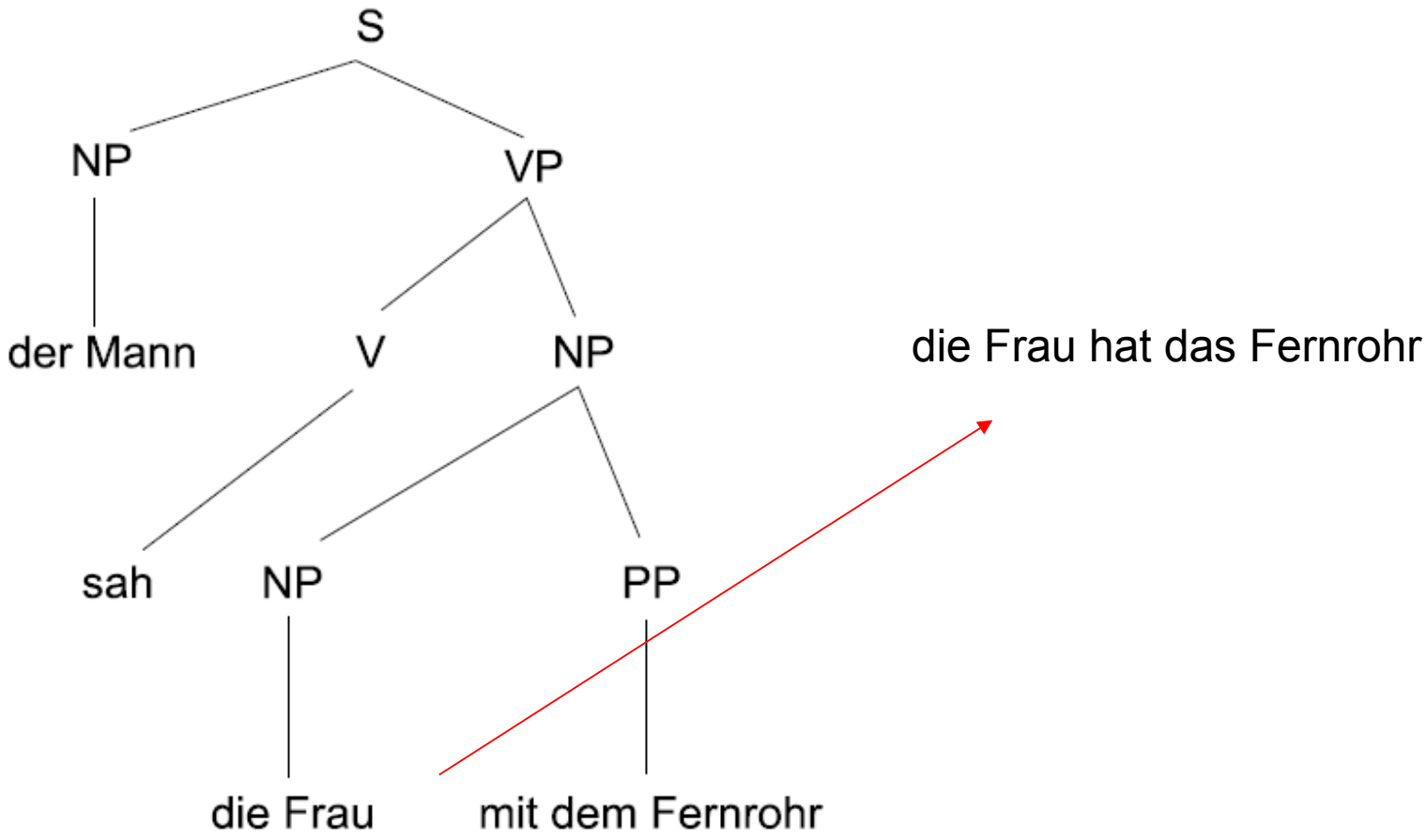
mehrere Tiefenstrukturen

Der Mann sah die Frau mit dem Fernrohr.



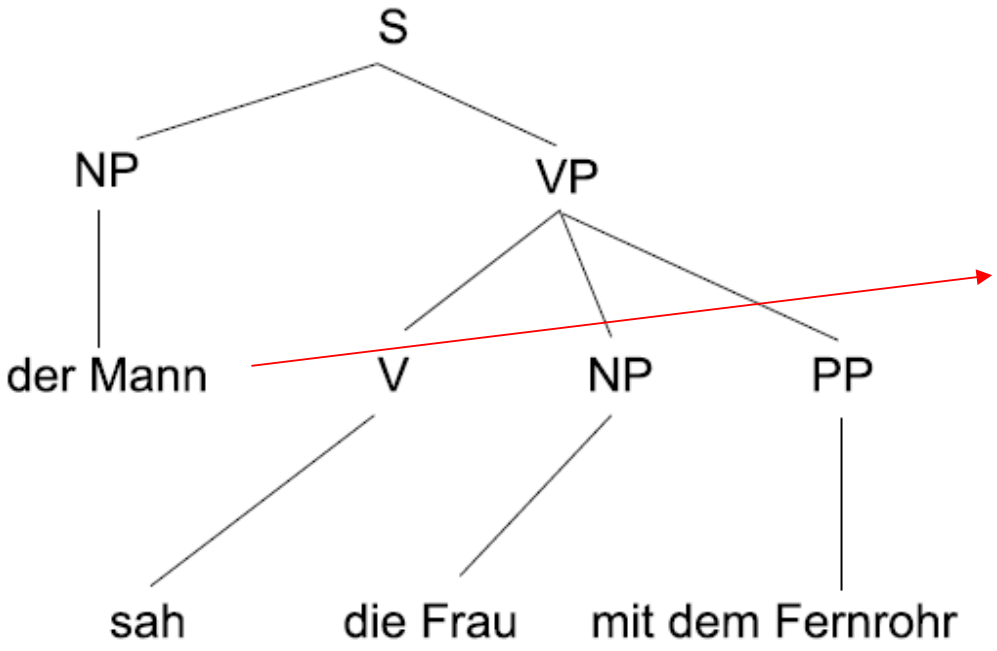
Tiefenstruktur 1

Der Mann sah die Frau mit dem Fernrohr.



Tiefenstruktur 2

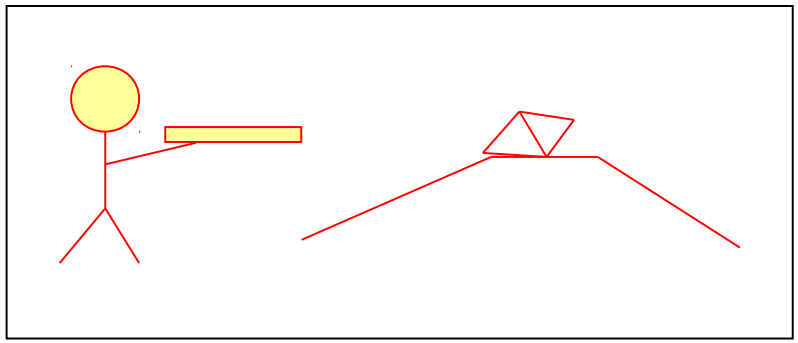
Der Mann sah die Frau mit dem Fernrohr.

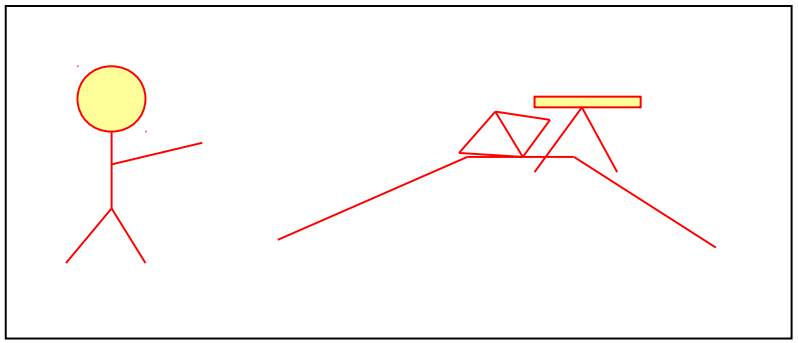


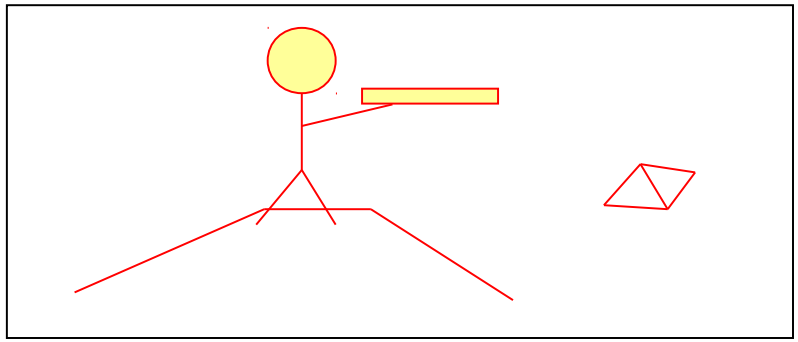
der Mann hat das Fernrohr

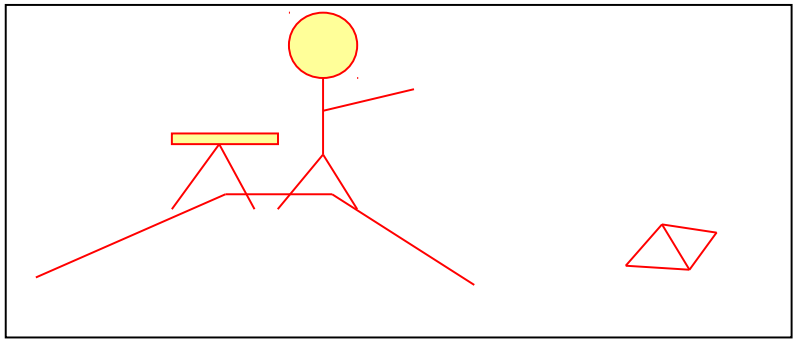
Weiteres Beispiel für mehrere Bedeutungen

Der Roboter sah mit einem Fernrohr die Pyramide auf dem Berg.









Beispiel 4

$$V_N = \{S, VP-PPS, ADJS-N, NP, VP, PP, N, V, P, ART, ADJ\}$$

$$V_T = \{Roboter, Pyramide, Tisch, \ldots, bewegte, \ldots,$$
$$rote, großen, törichte, graue, \ldots, der, die, \ldots, zum, \ldots\}$$

$$\begin{aligned}
S &\rightarrow NP \quad VP-PPS \\
NP &\rightarrow ART \quad ADJS-N \\
NP &\rightarrow ADJS-N \\
ADJS-N &\rightarrow ADJ \quad ADJS-N \\
ADJS-N &\rightarrow N \\
VP-PPS &\rightarrow VP-PPS \quad PP \\
VP-PPS &\rightarrow VP \\
VP &\rightarrow V \quad NP \\
VP &\rightarrow V \\
VP &\rightarrow V \quad ADJ \\
PP &\rightarrow P \quad NP \\
N &\rightarrow Roboter \mid Pyramide \mid Tisch \mid \ldots \\
V &\rightarrow bewegte \mid \ldots \\
ADJ &\rightarrow törichte \mid graue \mid rote \mid großen \mid \ldots \\
ART &\rightarrow der \mid die \mid \ldots \\
P &\rightarrow zum \mid \ldots
\end{aligned}$$

- ***PPS*** – Präpositionalphrasen
- ***ADJS*** – Adjektive
- ***VP-PPS*** – …
- ***ADJS-N*** – …

rekursive Regeln

Der törichte graue Roboter bewegte die rote Pyramide zum großen Tisch.

Beispiel 5 – Einbeziehung semantischer Elemente

$$V_N = \{S, V, P, AUSH, WZ, WSTEIL, WS, KOM\} \qquad V_T = \{bohre, fräse, \ldots\}$$

$$S \rightarrow V \left\{ \begin{array}{l} ein \\ eine \\ einen \end{array} \right\} AUSH \quad mit \left\{ \begin{array}{l} dem \\ der \end{array} \right\} WZ \quad P \left\{ \begin{array}{l} der \\ die \\ das \end{array} \right\} WSTEIL$$

$$V \rightarrow bohre, fräse, \ldots$$

$$AUSH \rightarrow Loch, Nut, Rille, \ldots$$

$$P \rightarrow in, amRand, entlang, durch, \ldots$$

$$WZ \rightarrow Bohrer, Meißel, Fräser, \ldots$$

$$WSTEIL \rightarrow KOM \left\{ \begin{array}{l} der \\ des \end{array} \right\} WS$$

$$KOM \rightarrow Flanke, Oberfläche, \ldots$$

$$WS \rightarrow Grundplatte, Holzbalken, \ldots$$

alternative Möglichkeiten

Bohre ein Loch mit dem Bohrer durch die Oberfläche der Grundplatte.

Fräse eine Nut mit dem Fräser in die Flanke des Holzbalkens.

4.2.3 Elementare Parsingalgoritmen

Parsingalgoritmen

- Der Vorgang der syntaktischen Verarbeitung eines Satzes (Symbolkette) unter Zuhilfenahme eines Regelwerkes (Grammatik) heißt Parsen.
- Prinzipiell lassen sich dabei zwei Hauptverfahren unterscheiden
  - Top – Down – Mechanismus
  - Bottom – Up – Mechanismus
- Weitere Verfahren
  - Left – corner – Analyse
  - Chart – Parser

Top – Down – Mechanismus

- Man beginnt beim Startsymbol und expandiert mit Hilfe der zur Verfügung stehenden Regeln solange, bis nur noch Terminalsymbole übrigbleiben.
- Diese Symbolkette wird dann mit dem zu analysierenden Satz verglichen.
- Wird eine Regelsequenz gefunden, welche das Startsymbol in dem zu analysierenden Satz umwandelt, dann begründet sie zusammen mit Informationen über die angewendeten Regeln eine Satzanalyse.

Bottom – Up – Mechanismus

- Man beginnt mit der Eingabesymbolfolge (zu analysierender Satz) und versucht, durch Anwendung der Regeln das Startsymbol der Grammatik rückwärts zu erreichen.

Beispiel 1

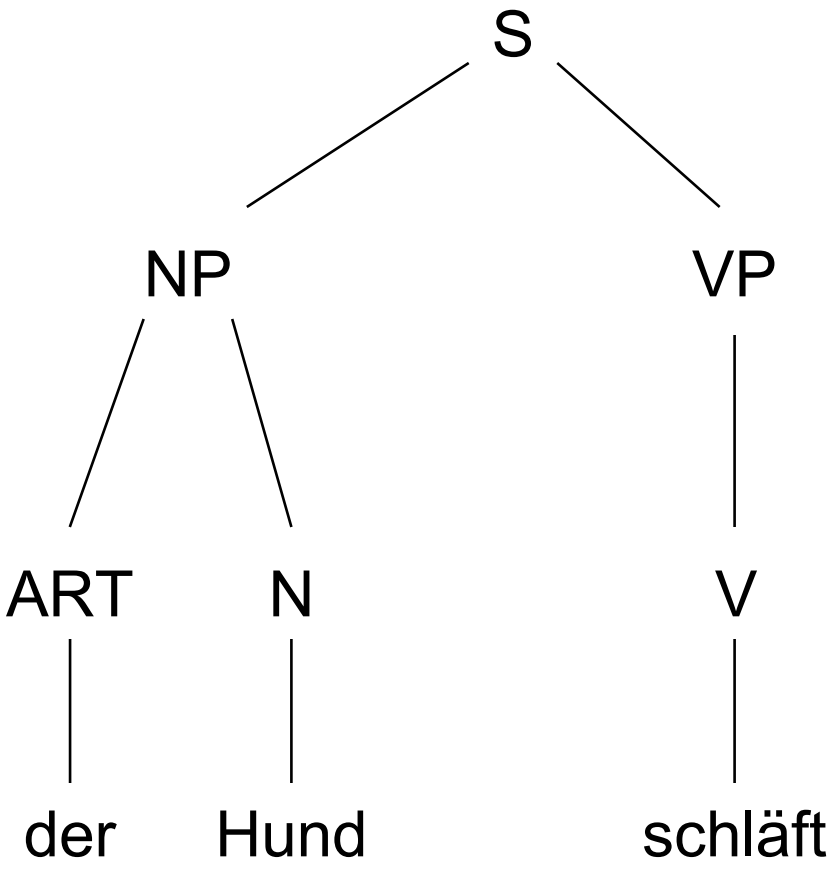
$V_N = \{S, NP, VP, N, V, ART\}$

$V_T = \{\text{Hund}, \text{schläft}, \text{der}\}$

Regeln:

S	→	NP VP
NP	→	N
NP	→	ART N
VP	→	V
N	→	Hund
V	→	schläft
ART	→	der

Beispiel 1



Parsing – Beispiel 1

Top – Down

S → NP VP

NP VP → ART N VP

ART N VP → der N VP

der N VP → der N V

der N V → der Hund V

der Hund V → der Hund schläft

Bottom – Up

der Hund schläft

der Hund V

der N V

ART N V

ART N VP

NP VP

S

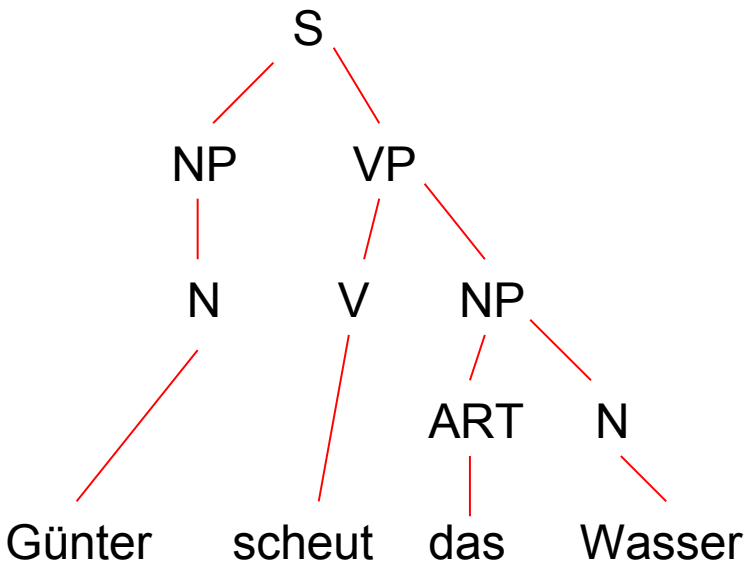
Top – Down: Beispiel 2

Günter scheut das Wasser.

$$
\begin{aligned}
S &\rightarrow NP\ VP \\
NP\ VP &\rightarrow N\ VP \\
N\ VP &\rightarrow Günter\ VP \\
Günter\ VP &\rightarrow Günter\ V\ NP \\
Günter\ V\ NP &\rightarrow Günter\ scheut\ NP \\
Günter\ scheut\ NP &\rightarrow Günter\ scheut\ ART\ N \\
Günter\ scheut\ ART\ N &\rightarrow Günter\ scheut\ das\ N \\
Günter\ scheut\ das\ N &\rightarrow Günter\ scheut\ das\ Wasser
\end{aligned}
$$

Top – Down: Beispiel 2

Günter scheut das Wasser.



Bottom – Up: Beispiel 4

Der törichte graue Roboter bewegte die rote Pyramide zum großen Tisch
ART *törichte graue Roboter bewegte die rote Pyramide zum großen Tisch*
ART **ADJ** *graue Roboter bewegte die rote Pyramide zum großen Tisch*
ART ADJ **ADJ** *Roboter bewegte die rote Pyramide zum großen Tisch*
ART ADJ ADJ **N** *bewegte die rote Pyramide zum großen Tisch*
ART ADJ ADJ **ADJS-N** *bewegte die rote Pyramide zum großen Tisch*
ART ADJ **ADJS-N** *bewegte die rote Pyramide zum großen Tisch*
ART **ADJS-N** *bewegte die rote Pyramide zum großen Tisch*
NP *bewegte die rote Pyramide zum großen Tisch*
NP **V** *die rote Pyramide zum großen Tisch*
NP V **ART** *rote Pyramide zum großen Tisch*
NP V ART **ADJ** *Pyramide zum großen Tisch*
NP V ART ADJ **N** *zum großen Tisch*
NP V ART ADJ **ADJS-N** *zum großen Tisch*
NP V ART **ADJS-N** *zum großen Tisch*
NP V **NP** *zum großen Tisch*
NP **VP** *zum großen Tisch*
NP **VP-PPS** *zum großen Tisch*
NP VP-PPS **P** *großen Tisch*
NP VP-PPS P **ADJ** *Tisch*
NP VP-PPS P ADJ **N**
NP VP-PPS P ADJ **ADJS-N**
NP VP-PPS P **ADJS-N**
NP VP-PPS P **NP**
NP VP-PPS **PP**
NP **VP-PPS**
S

Der törichte graue Roboter bewegte die rote Pyramide zum großen Tisch.

Left – corner – Analyse

- Algorithmen dieses Typs verwenden eine gemischte Strategie.
- Die erste Teilkonstituente (linke Ecke - left corner) jeder Konstituente wird bottom – up erkannt, alle übrigen Teilkonstituenten top – down.

Beispiel

Günter scheut das Wasser.

$S \,/\, NP\ VP$
$Günter$
N
NP
$S \,/\, VP$
$scheut$
V
$VP \,/\, NP$
das
ART
$NP \,/\, N$
$Wasser$
N

$$A/B_1 B_2 \cdots B_n$$

Um eine Konstituente des Typs A zu erkennen, müssen noch die Konstituenten B_1 … B_n erkannt werden.

Sie formulieren also Top – down Erwartungen des Parsers.

Probleme

- Syntax und Lexikon müssen eine deterministische Analyse des Satzes erlauben (in jedem Schritt kann genau eine Regel angewendet werden). Dies ist i. a. nicht erfüllt. Deshalb verwendet man Suchstrategien (Tiefensuche, Breitensuche) um Alternativen zu verfolgen.
- Nur eingeschränkte Klassen von Grammatiken können verarbeitet werden.
  - keine links- / rechtsrekursiven Regeln bei Top – down – Parsing
  - keine Tilgungsregeln bei Bottom – up – Parsing und Left – corner – Parsing.
- Die Algorithmen sind ineffizient. In vielen Fällen wird ein Satzabschnitt mehrfach auf dieselbe Art und Weise analysiert.

4.2.4 Chart-Parser

(J. Earley, 1967)

Chart – Parser

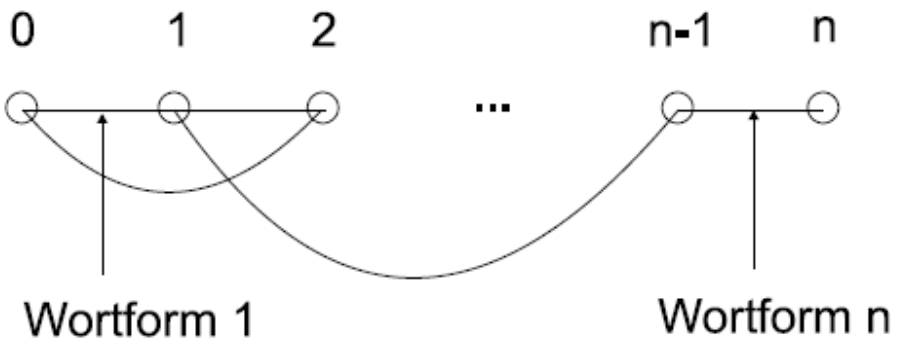
- Es müssen Verfahren gefunden werden, die es erlauben, Mehrfachanalysen zu vermeiden.
- Ein Ansatz besteht darin, die bei der voranschreitenden Analyse gewonnenen Teilergebnisse zu speichern.
- Auf diese kann dann zugegriffen werden, wenn für den Satz eine andere Analysemöglichkeit als die zunächst verfolgte geprüft wird.
- Zur Speicherung der Analyseergebnisse wird häufig ein Chart verwendet.
- Ein Chart kann als Graph aufgefasst werden.

Chart

Satz $S \rightarrow \ w_1 w_2 \cdots w_n$

Wortform w_i

Chart



Kanten

- aktive Kanten
  - repräsentieren eine partielle Analyse
- passive Kanten
  - repräsentieren eine abgeschlossene Analyse
- bei der syntaktischen Analyse eines Satzes werden schrittweise mit Hilfe der Grammatikregeln immer neue Kanten erzeugt
- Aufbau einer Kante *K*

[Start(*K*), Ende(*K*), Kopf(*K*), geschlossener_Abschnitt(*K*), offener_Abschnitt(*K*)]

Anfangsknoten der Kante

Endknoten der Kante

Typ der durch *K* repräsentierten Konstituente (z.B.: S, NP, ...)

analysierter Teil der durch *K* repräsentierten Konstituente

noch nicht analysierter Teil der durch *K* repräsentierten Konstituente

Beispiele

Initialisierung (der gesamte Satz S ist noch zu analysieren)

$$[0,0,S,\varepsilon\ ,NP\,VP]$$

Regel: $S \rightarrow\ NP\,VP$

geschlossener Abschnitt ist leer

Ziel ist die passive Kante
(der gesamte Satz S ist analysiert)

$$[0,n,S,NP\,VP,\varepsilon\ ]$$

offener Abschnitt ist leer

Prozeduren zur Berechnung neuer Kanten

- EXPAND
  - Hüllenbildung
- SCAN
  - Integration eines Terminalsymbols
- COMPLETE
  - Kombination zweier Teilkonstituenten

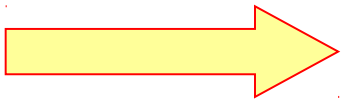
EXPAND

Gegeben:
- kontextfreie Grammatik G
- Chart C

$$G = \left(V_N, V_T, R, S\right)$$

$$\left[i, j, A, \alpha\ , B\ \beta\ \right] \in\ C \qquad 0 \leq i \leq j \leq n,\ \ A, B \in V_N,\ \ \alpha\ , \beta\ \in V^*$$

$$\left(B \rightarrow\ \tau\ \right) \in\ R \qquad \tau \in V^*$$

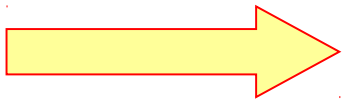


$$\left[j, j, B, \varepsilon\ , \tau\ \right] \in\ C$$

neue Kante

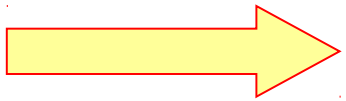
Beispiel

$[0,0,S,\varepsilon\ ,NP\ VP]$ $\qquad\qquad$ $NP \rightarrow\ \ N$



$[0,0,NP,\varepsilon\ ,N]$

$[0,0,S,\varepsilon\ ,NP\ VP]$ $\qquad\qquad$ $NP \rightarrow\ \ ART\ N$



$[0,0,NP,\varepsilon\ ,ART\ N]$

SCAN

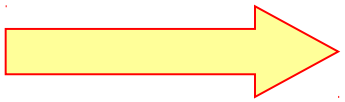
Gegeben:
- kontextfreie Grammatik G
- Chart C
- zu analysierender Satz S

$$G = \left(V_N, V_T, R, S\right)$$

$$S \rightarrow \; w_1 w_2 \cdots w_n$$

$$\left[i, j-1, A, \alpha \;, K\; \beta\;\right] \in \; C \qquad 0 \le i \le j-1 \le n, \;\; A, K \in V_N, \;\; \alpha\;, \beta \in V^*$$

$$\left(K \rightarrow \; w_j\right) \in \; R \qquad w_j \in V_T$$

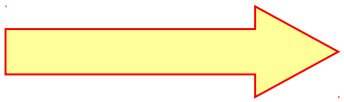


neue Kante

$$\left[i, j, A, \alpha \;\; K, \beta\;\right] \in \; C$$

Beispiel

$[0,0,NP,\varepsilon\ ,N]$ $\qquad\qquad$ $N \rightarrow\ \ w_1$



$[0,1,NP,N,\varepsilon\ ]$

COMPLETE

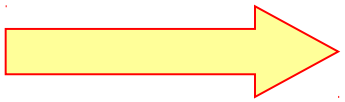
Gegeben:
- kontextfreie Grammatik G
- Chart C

$$G = \left(V_N, V_T, R, S\right)$$

$$\left[i, j, A, \alpha\ , B\ \beta\ \right] \in\ C$$

$$\left[\, j, k, B, \tau\ , \varepsilon\ \right] \in\ C$$

$$0 \le i \le\ j \le\ k \le\ n,\ \ A, B \in\ V_N,\ \ \alpha\ , \beta\ , \tau\ \in\ V^*$$



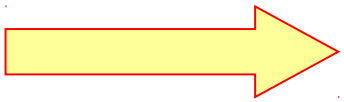
neue Kante

$$\left[i, k, A, \alpha\ \ B, \beta\ \right] \in\ C$$

Beispiel

$[0,0,S,\varepsilon\ ,NP\,VP]$

$[0,1,NP,N,\varepsilon\ ]$



$[0,1,S,NP,VP]$

Berechnung des Chart

- initialisiere den Chart

$$[0,0,S,\varepsilon\ ,NP\ VP]$$

- wende die Prozeduren (z.B. in folgender Reihenfolge)
  - SCAN
  - EXPAND
  - COMPLETE

  solange an, bis keine weiteren Kanten mehr erzeugt werden können
- Ist

$$[0,n,S,NP\ VP,\varepsilon\ ] \in\ C\,?$$

Beispiel

$G = (V_N, V_T, R, S)$

$$V_N = \{S, NP, VP, N, V, ART, PP, P\}$$

$$V_T = \{Hans, suchte, Peter, \ldots\}$$

$$\begin{aligned}
S &\rightarrow NP\ VP \\
NP &\rightarrow N \\
NP &\rightarrow ART\ N \\
VP &\rightarrow V\ NP \\
VP &\rightarrow V\ NP\ PP \\
PP &\rightarrow P\ NP \\
N &\rightarrow Hans \\
N &\rightarrow Peter \\
V &\rightarrow suchte \\
&\ldots
\end{aligned}$$

Hans suchte Peter.

Nr.	neue Kante	Prozedur	Kanten/Regeln
k1	$[0,0,S,\varepsilon,NP\ VP]$	INIT	$S \rightarrow NP\ VP$
k2	$[0,0,NP,\varepsilon,N]$	EXPAND	$NP \rightarrow N$
k3	$[0,0,NP,\varepsilon,ART\ N]$	EXPAND	$NP \rightarrow ART\ N$
k4	$[0,1,NP,N,\varepsilon]$	SCAN	k2
k5	$[0,1,S,NP,VP]$	COMPLETE	k1,k4
k6	$[1,1,VP,\varepsilon,V\ NP\ PP]$	EXPAND	$VP \rightarrow V\ NP\ PP$
k7	$[1,1,VP,\varepsilon,V\ NP]$	EXPAND	$VP \rightarrow V\ NP$
k8	$[1,2,VP,V,NP\ PP]$	SCAN	k6
k9	$[2,2,NP,\varepsilon,N]$	EXPAND	$NP \rightarrow N$
k10	$[2,2,NP,\varepsilon,ART\ N]$	EXPAND	$NP \rightarrow ART\ N$
k11	$[1,2,VP,V,NP]$	SCAN	k7
k12	$[2,3,NP,N,\varepsilon]$	SCAN	k9
k13	$[1,3,VP,V\ NP,\varepsilon]$	COMPLETE	k11,k12
k14	$[0,3,S,NP\ VP,\varepsilon]$	COMPLETE	k5,k13
k15	$[1,3,VP,V\ NP,PP]$	COMPLETE	k8,k12
k16	$[3,3,PP,\varepsilon,P\ NP]$	EXPAND	$PP \rightarrow P\ NP$

4.2.5 ID/LP – Grammatiken

ID – immediate dominance

LP – linear precedence

Idee

- Genau besehen enthalten die Regeln einer Phrasenstrukturgrammatik zwei unterschiedliche Arten von Informationen:
  - Dominanzinformation: z.B. ***NP*** besteht aus ***ART, ADJ*** und ***N*** (ID)
  - Präzedenzinformation: z.B. ***ART*** kommt vor ***ADJ***, ***ADJ*** vor ***N*** (LP)
- Diese beiden Aspekte von Phrasenstrukturregeln können auch entkoppelt werden.
- Man erhält dann zwei disjunkte Mengen von Regeln.

Definition

$$G = (V_N, V_T, D, P, S)$$

V	endliche Menge von Symbolen (Alphabet)		
V_N	nichtterminale Symbole	$V_N \subseteq V$	$V_N \cup V_T = V$
V_T	terminale Symbole	$V_T \subseteq V$	$V_N \cap V_T = \emptyset$
S	Startsymbol	$S \in V_N$	

Definition

$$G = \left(V_N, V_T, D, P, S\right)$$

$D \subseteq \left(V^* \setminus V_T^*\right) \times V^*$ endliche Menge von Regeln

$r \in D \qquad r = \left(\varphi, \psi\right) \qquad r : \varphi \rightarrow \psi \qquad \varphi \neq \varepsilon$

$P \subseteq V \times V$ endliche Menge von Ordnungsrelationen

$r \in P \qquad r = \left(\varphi, \psi\right) \qquad r : \phi \prec \psi$

$\exists\, r = \left(S, \psi\right) \in D$

Bemerkungen

- Dominanzregeln geben an, aus welchen Subphrasen Phrasen bestehen.
- Präzedenzregeln beschreiben eine partielle Ordnung der Phrasen.
- Die Präzedenzregeln haben über der gesamten Menge der Dominanzregeln Gültigkeit.

Beispiel

$$S \rightarrow \; a\, b\, c \qquad\qquad b \lessdot c$$

Als kontextfreie Grammatik:

$$S \rightarrow \; a\, b\, c$$

$$S \rightarrow \; b\, c\, a$$

$$S \rightarrow \; b\, a\, c$$

Äquivalente Grammatiken

G_1 schwach äquivalent G_2

$$L_{G_1} = L_{G_2}$$

G_1 stark äquivalent G_2

$L_{G_1} = L_{G_2}$ *und beide Grammatiken* allen *Sätzen identische Strukturbeschreibungen zuordnen*

Zusammenhang zu kontextfreien Grammatiken

- zu jeder ID/LP – Grammatik gibt es eine stark äquivalente kontextfreie Grammatik (siehe obiges Beispiel)
- für jede kontextfreie Grammatik kann eine schwach äquivalente ID/LP – Grammatik angegeben werden

Beispiel

kontextfreien Grammatik:

$$\begin{aligned} X_1 &\rightarrow Y_1 \quad Y_2 \\ X_2 &\rightarrow Y_2 \quad Y_1 \end{aligned}$$

schwach Äaquivalente ID/LP – Grammatik:

$$\begin{aligned} ID - Regel: &\quad X_1 \rightarrow Z_1 \quad Y_2 \\ ID - Regel: &\quad X_2 \rightarrow Y_2 \quad Z_2 \\ ID - Regel: &\quad Z_1 \rightarrow Y_1 \\ ID - Regel: &\quad Z_2 \rightarrow Y_1 \\ LP - Regel: &\quad Z_1 < Y_2 \\ LP - Regel: &\quad Y_2 < Z_2 \end{aligned}$$

Beispiel

kontextfreien Grammatik:

$$A \rightarrow B \quad C$$
$$D \rightarrow E \quad C \quad B$$

schwach äquivalente ID/LP – Grammatik:

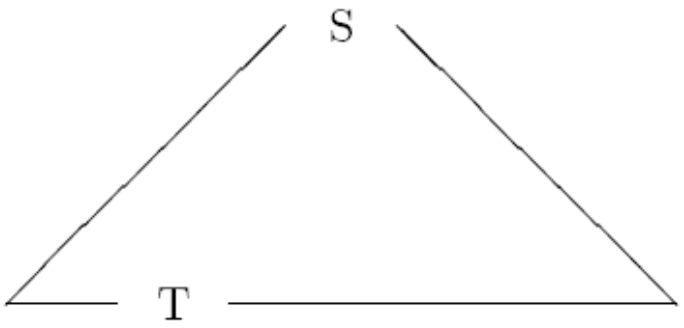
$$ID - Regel: \quad A \rightarrow B_1 \quad C_1$$
$$ID - Regel: \quad D \rightarrow E \quad C_2 \quad B_2$$
$$ID - Regel: \quad B_1 \rightarrow B$$
$$ID - Regel: \quad C_1 \rightarrow C$$
$$ID - Regel: \quad B_2 \rightarrow B$$
$$ID - Regel: \quad C_2 \rightarrow C$$
$$LP - Regel: \quad B_1 < C_1$$
$$LP - Regel: \quad E < C_2$$
$$LP - Regel: \quad C_2 < B_2$$

4.2.6 Baumadjunktionsgrammatiken

Tree Adjoining Grammar – TAG

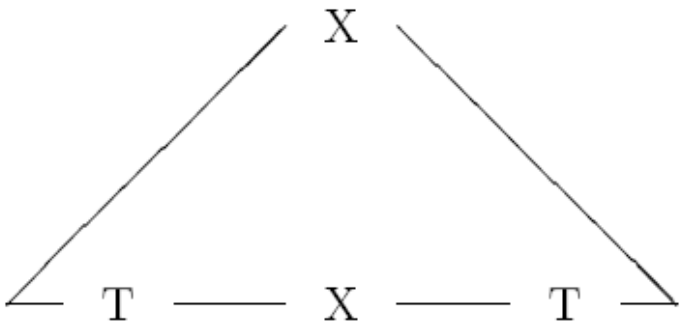
Initialbaum

- alle Knoten gehören zur Menge V
- *die Wurzel ist gleich dem Startsymbol* S
- *alle Blätter sind Elemente der Menge* V_T



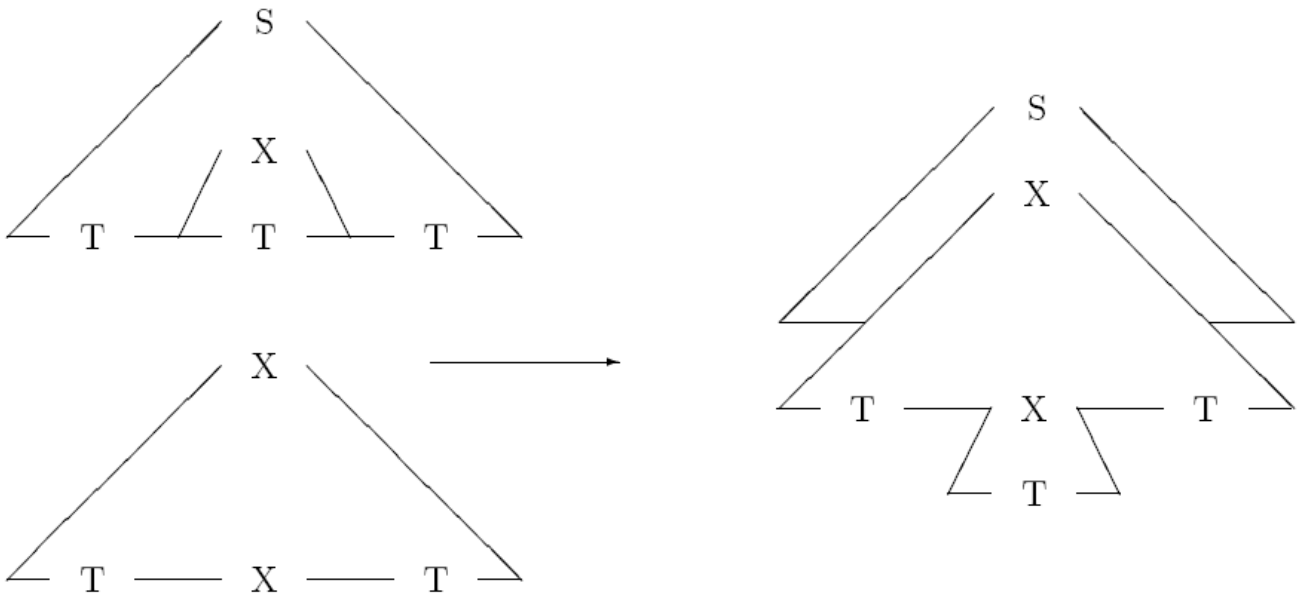
Auxiliarbaum

- alle Knoten gehören zur Menge V
- alle Blätter sind Elemente der Menge V_T bis auf genau ein nichtterminales Symbol X
- *die Wurzel ist gleich diesem Symbol* X



Adjunktion

Adjunktion *ist eine Operation, bei der man jeden Auxiliarbaum an Stelle eines beliebigen Teilbaums in einen anderen Baum einsetzen kann, sofern dieser Teilbaum dieselbe Wurzel wie der Auxiliarbaum hat. Der Teilbaum selbst wird dann seinerseits anstelle des nichtterminalen Blattes X im Auxiliarbaum eingesetzt.*



TAG

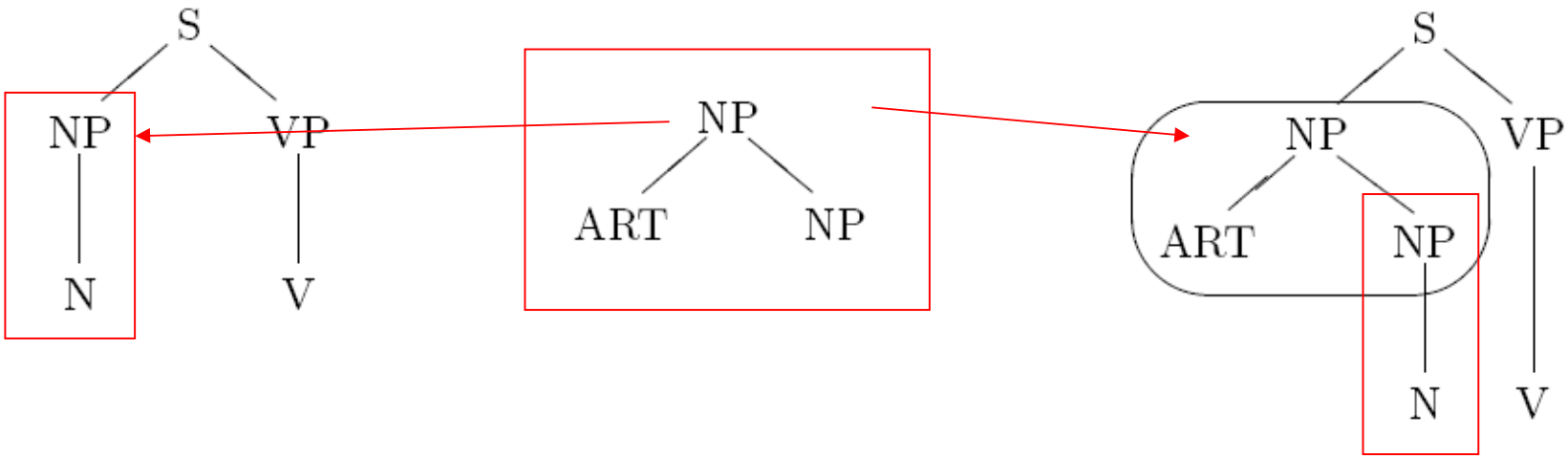
$$G = (V_N, V_T, I, A, J, S)$$

V	endliche Menge von Symbolen (Alphabet)		
V_N	nichtterminale Symbole	$V_N \subseteq V$	$V_N \cup V_T = V$
V_T	terminale Symbole	$V_T \subseteq V$	$V_N \cap V_T = \emptyset$
S	Startsymbol	$S \in V_N$	
I	endliche Menge von Initialbäumen		
A	endliche Menge von Auxiliarbäumen		

$J : (\{\text{Menge der abgeleiteten Bäume}\} \cup I) \rightarrow \{\text{Menge der abgeleiteten Bäume}\}$

Adjunktion ist eine Abbildung zur Erzeugung abgeleiteter Bäume

Beispiel



Initialbaum

Auxiliarbaum

Adjunktion

Bemerkungen

- jeder Initialbaum repräsentiert einen gültigen Satz der Sprache
- durch Adjunktion werden ebenfalls stets Bäume erzeugt, die gültige Sätze repräsentieren
- alle kontextfreien Sprachen können mit TAGs dargestellt werden, aber nicht umgekehrt
- TAGs gehen über die Mächtigkeit von kontextfreien Sprachen hinaus
- darüber hinaus können TAGs auch eine bestimmte Klasse kontextsensitiver Sprachen akzeptieren, nämlich die Klasse der schwach kontextsensitive Sprachen
- es wird die Hypothese vertreten, dass die Komplexität von TAGs genau der Syntax der natürlichen Sprachen entspricht

4.2.7 Probleme

Probleme

- Eine Grammatik beschreibt nur entweder eine Unter- oder eine Obermenge einer natürlichen Sprache.
- Manche linguistischen Daten sind mit Regeln nur schwer darstellbar (Fernabhängigkeiten, Übereinstimmung)

NP ⟶ ART ADJ N

der alter Vater

ein junges Mann