Probabilistische kontextfreie Grammatiken und Parsing

Sebastian Pado

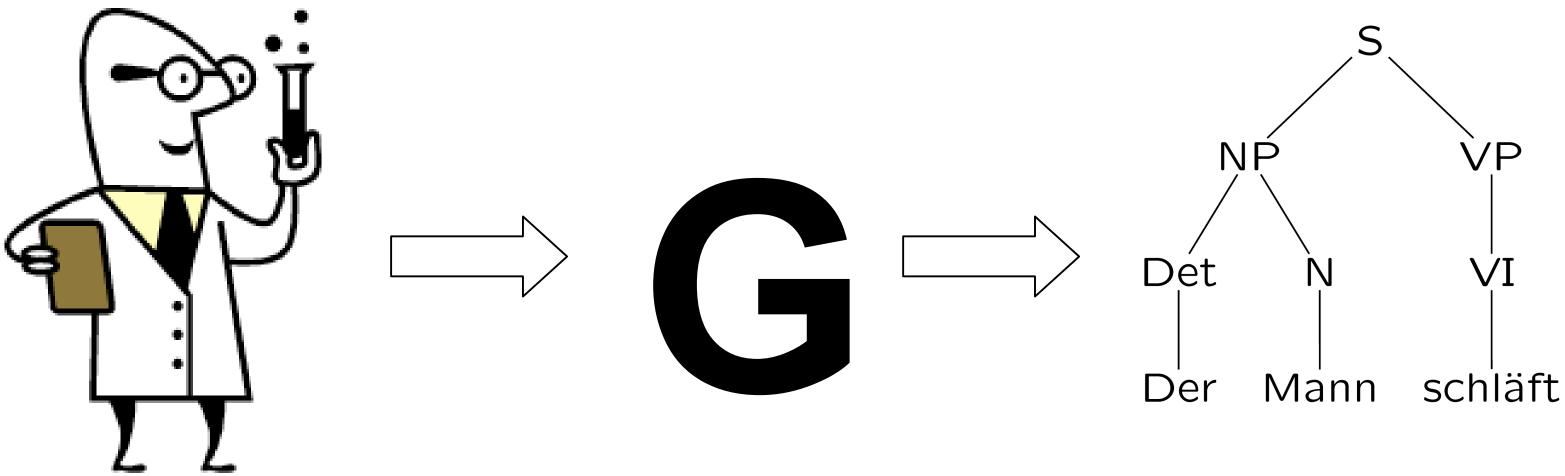
18.01.2005

Robustes Parsing

- Ziel: Syntaktische Analyse von freiem Text
- Anwendungen:
  - Freier Dialog
  - Große Textmengen (Internet)
- Herausforderungen für Grammatik:
  - Fehler durch falsche Spracherkennung
  - Unterschied zwischen Grammatik und Gebrauch
    - Umgangssprachliche Konstruktionen

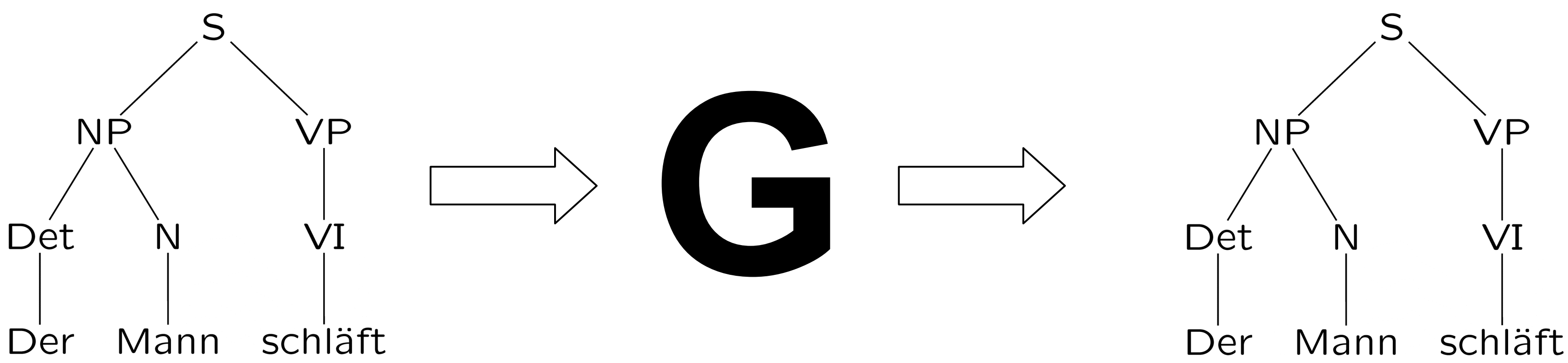
Erzeugung einer großen Grammatik

- Von Hand schreiben? ☹
  - Problem: Überblick
  - Problem: Modellierung von Umgangssprache



Erzeugung einer großen Grammatik

- Stattdessen: Betrachte **Baumbank**
  - Korpus von Sätzen mit syntaktischer Analyse
    - Manuelle Annotation: Hohe Genauigkeit
  - Enthält implizites linguistisches Wissen
    - Kann in Form einer Grammatik extrahiert werden
    - Enthält automatisch alle gebrauchten Phänomene



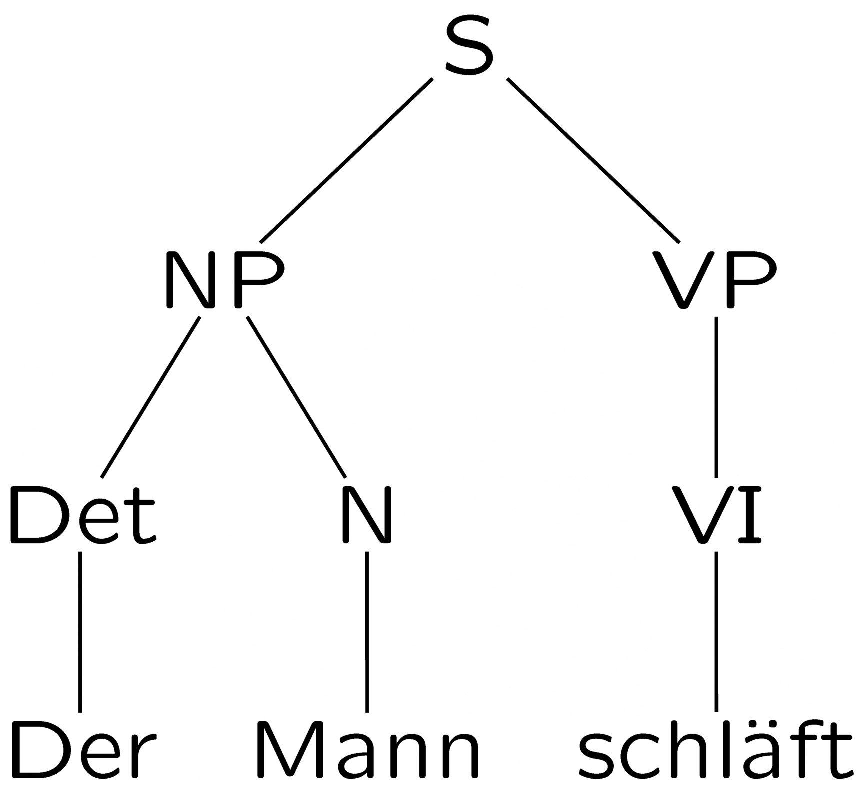
Extraktion einer Baumbank-Grammatik

Baum = Menge von Regeln

- Naiver Extraktionsalgorithmus einer kontextfreien Baumbank-Grammatik:
  - Initialisierung: „leere Grammatik“
  - Eine (kontextfreie) Baumbank nehmen
  - Alle Parsebäume in „lokale Bäume“ aufteilen
  - Jeden lokalen Baum als Regel zur Grammatik hinzufügen

Naive Baumbank-Grammatiken

Parsebaum



Abgeleitete Regeln

S -> NP VP
NP -> Det N
Det -> Der
N -> Mann
VP -> VI
VI -> schläft

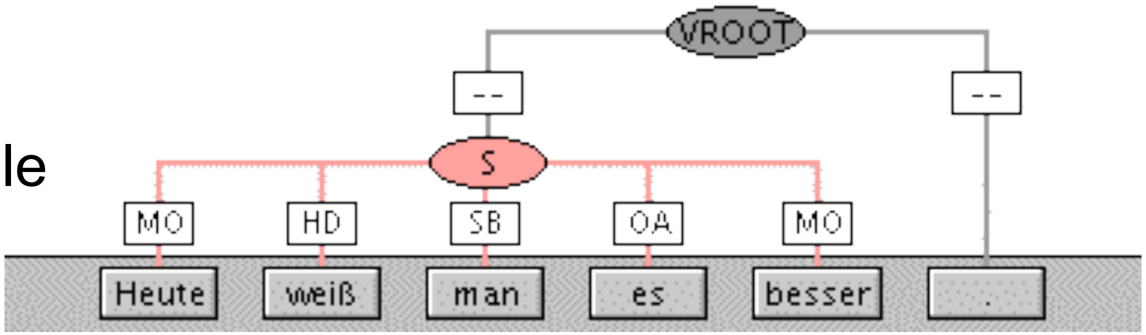
Probleme von Baumbank-Grammatiken

1. Form der Baumbank-Regeln

Gelernte Regeln sind vom **Annotationsschema** der annotierten Baumbank abhängig

- Tiefere Bäume
  - mehr Nonterminale
  - weniger Regeln

- Flachere Bäume
  - weniger Nonterminale
  - mehr Regeln



Kontextfreie Baumbank-Grammatiken sind sehr flach

Tiefe und flache Grammatiken

- Flache Grammatik: **Viele** Regeln („kombinatorische Explosion“)
  - Beispiel: Regeln für NPs mit und ohne AdjP

Traditionelle Grammatik
(mit AdjP)

NP -> AdjP N
NP -> Det AdjP N

AdjP -> ε
AdjP -> Adj AdjP

Baumbank-Grammatik
(ohne AdjP)

NP -> N
NP -> Adj N
NP -> Adj Adj N

NP -> Det N
NP -> Det Adj N
NP -> Det Adj Adj N

Flache Regeln (dt. Baumbank NEGRA)

...
NP -> ADJA $*LRB* $, ADJA NN
NP -> ADJA $*LRB* NN
NP -> ADJA $*LRB* NN $*LRB* PP
NP -> ADJA $*LRB* NN PP
NP -> ADJA $, ADJA NN
NP -> ADJA $, ADJA NN PP
NP -> ADJA $, AP NN
NP -> ADJA ADJA ADJA NN
NP -> ADJA ADJA CARD
NP -> ADJA ADJA CNP
NP -> ADJA ADJA NN $*LRB* NE
NP -> ADJA ADJA NN $*LRB* NP
NP -> ADJA ADJA NN NP
NP -> ADJA ADJA NN
...

$,: Komma
ADJA: Attributives Adjektiv
CARD: Kardinalzahl
CNP: Koordinierte NP
$*LRB*: Klammer
NE: Eigennamen
NN: Normales Nomen
NP: Nominalphrase
PP: Präpositionalphrase

Grammatik wird sehr groß

2. Mehrdeutigkeit

- Große Grammatiken enthalten sehr viele seltene Regeln
  - „Zipfsche Verteilung“
- Wieso ist das problematisch?
  - Seltene Regeln enthalten u.U. sehr häufige Muster
    - Fehlanalysen
- Beispiel:
  - Kleine Gruppe von Verben nimmt mehrere PP-Objekte
    - Bewegungsverben
    - Die Aktien **steigen [von 50 Euro] [um 10 Euro] [auf 60 Euro]**
  - Grammatik enthält Regel VP -> V PP PP PP

Mehrdeutigkeit

- Auch auf Sätze, die gar nicht so analysiert werden können, wird diese Regeln angewendet:

  Peter **telefoniert [mit dem Handy] [von Siemens] [in der Hand]**
  Ich **stehe [mit dem Schal] [auf der Brücke] [von Köln]**
  VP -> V PP PP PP ?????

- Resultat: Eigentlich eindeutige Sätze werden mehrdeutig
  - Sätze mit 20-30 Worten erhalten z.T. > 1 Mio. Analysen
    - Verarbeitungsproblem
    - Interpretationsproblem

3. Linguistische Relevanz

- Grammatik als linguistisches Modell:
  1. Grammatikalität eines Satzes
  2. Syntaktische Struktur eines Satzes
- Gute Grammatik macht richtige Aussagen

- Kleine Grammatik = besseres Modell
  - Bessere Analysen für neue Sätze
  - Occams Razor

Probleme mit der Grammatikalität

- AdjP-Beispiel

Grammatik 1

NP -> AdjP N
NP -> Det AdjP N

AdjP -> ε
AdjP -> Adj AdjP

Grammatik 2

NP -> N
NP -> Adj N
NP -> Adj Adj N

NP -> Det N
NP -> Det Adj N
NP -> Det Adj Adj N

- Analyse von “Alte große grüne Häuser”?
  - G2 erkennt NP nicht als grammatisch: Kein gutes Modell

Baumbank-Grammatiken

- Baumbank-Grammatiken sehr groß
  - 19.000 Regeln für 18.000 Sätze NEGRA-Korpus
  - Fast jede Kette von Wörtern wird als Satz erkannt
    - Kein gutes Modell
      - Kann aber als „robustes“ Anwendungsmodell gewuenscht sein
  - Viele Sätze erhalten sehr viele Analysen
    - Kein gutes Model

4. Grammatikalität

- Naive Baumbank-Grammatiken haben kein Konzept von **gradueller** Grammatikalität
  - Keine Möglichkeit, von plausibleren / unplausibleren Analyse zu sprechen
  - Besonders schlimm wegen hoher Mehrdeutigkeit
    - Was nutzen 1 Mio. Analysen für einen Satz?

Verbesserung: Probabilistische kontextfreie Grammatiken (PCFGs)

Probabilistische Grammatiken (PCFGs)

- Jede Regel erhält eine Wahrscheinlichkeit
  - S -> NP VP (0.6)
  - S -> NP VP PP (0.4)
    - „60% aller Sätze haben die Form NP VP, 40% die Form NP VP PP“
- Wahrscheinlichkeiten für Parsebäume
  - **Berechnung aus Regelwahrscheinlichkeiten**
- Vorteile
  - Modellseite: Graduelle Grammatikalität
    - Beste Analyse = wahrscheinlichster Parsebaum
  - Verarbeitungsseite: Einschränkung des Suchraumes
    - Verfolge vor allem „wahrscheinliche“ Parsebäume

Parsen mit PCFGs

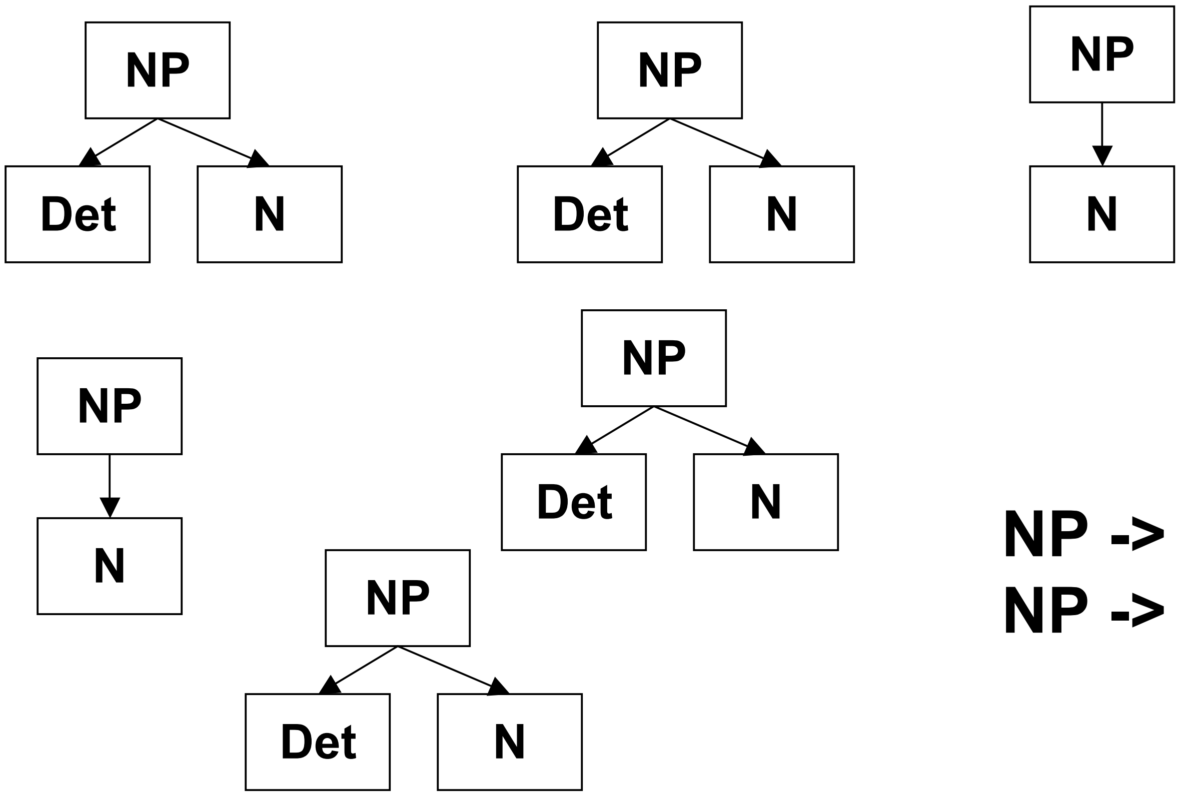
1. Lerne Wahrscheinlichkeiten für Regeln
2. Berechne Wahrscheinlichkeiten für Bäume aus Regelwahrscheinlichkeiten
3. (Finde effizientes Verfahren, um wahrscheinlichste Bäume zu konstruieren)

Regelwahrscheinlichkeiten lernen

- Nicht naive Baumbank-Grammatik:
  1. Baumbank nehmen
  2. Leere Grammatik initialisieren
  3. Alle Parsebäume in „lokale Bäume“ aufteilen
  4. Jeden neuen lokalen Baum als Regel hinzufügen
     - Zähle (relative) Häufigkeit von Regeln

- Terminologie:
  - Linke Seite (LS): Nonterminal
  - Rechte Seite (RS): Expansion

Beispiel



NP -> N 2 von 6
NP -> Det N 4 von 6

Regelwahrscheinlichkeiten

- Gleiche Idee wie bei statistischer Klassifikation (letzte Vorlesung)
  - Dort: Für gegebenes Ereignis, sammle Frequenzen der Klassen
  - Hier: Für gegebenes Nonterminal, sammle Frequenzen der Expansionen:

Von allen Auftreten des Nonterminals,
wie oft kommt eine Expansion vor?

- Schreibweise: P(Expansion | Nonterminal)
  - P = probability (Wahrscheinlichkeit)
  - Beispiel von letzter Seite:
    - P(N | NP) = 2/6
    - P(Det N | NP) = 4/6

„Echte“ Regelfrequenzen in NEGRA

...	
1	NP ADJA $*LRB* $, ADJA NN
12	NP ADJA $*LRB* NN
1	NP ADJA $*LRB* NN $*LRB* PP
2	NP ADJA $*LRB* NN PP
8	NP ADJA $, ADJA NN
1	NP ADJA $, ADJA NN PP
3	NP ADJA $, AP NN
1	NP ADJA ADJA ADJA NN
1	NP ADJA ADJA CARD
2	NP ADJA ADJA CNP
1	NP ADJA ADJA NN $*LRB* NE
1	NP ADJA ADJA NN $*LRB* NP
4	NP ADJA ADJA NN NP
51	NP ADJA ADJA NN
...	

P(ADJA LRB NN | NP) = 12/89

P(ADJA ADJA NN | NN) = 51/89

P(ADJA ADJA CARD | NP) = 1/89

Parsing mit PCFGs

1. Lerne Regeln mit Wahrscheinlichkeiten

2. Berechne Wahrscheinlichkeiten für Bäume aus Regelwahrscheinlichkeiten

3. (Finde effizientes Verfahren, um wahrscheinlichste Bäume zu konstruieren)

Baumwahrscheinlichkeiten berechnen

- Baum ist Menge an Regeln
  - Z.B. (R_1, R_2, R_3)
- Bekannt: Wahrscheinlichkeit der Regeln
  - $P(R_1), P(R_2), P(R_3)$
- Gesucht: Wahrscheinlichkeit des Baumes
  - Idee: Produkt der Regelwahrscheinlichkeiten
    $P(R_1, R_2, R_3) = \prod_i P(R_i) = P(R_1) * P(R_2) * P(R_3)$

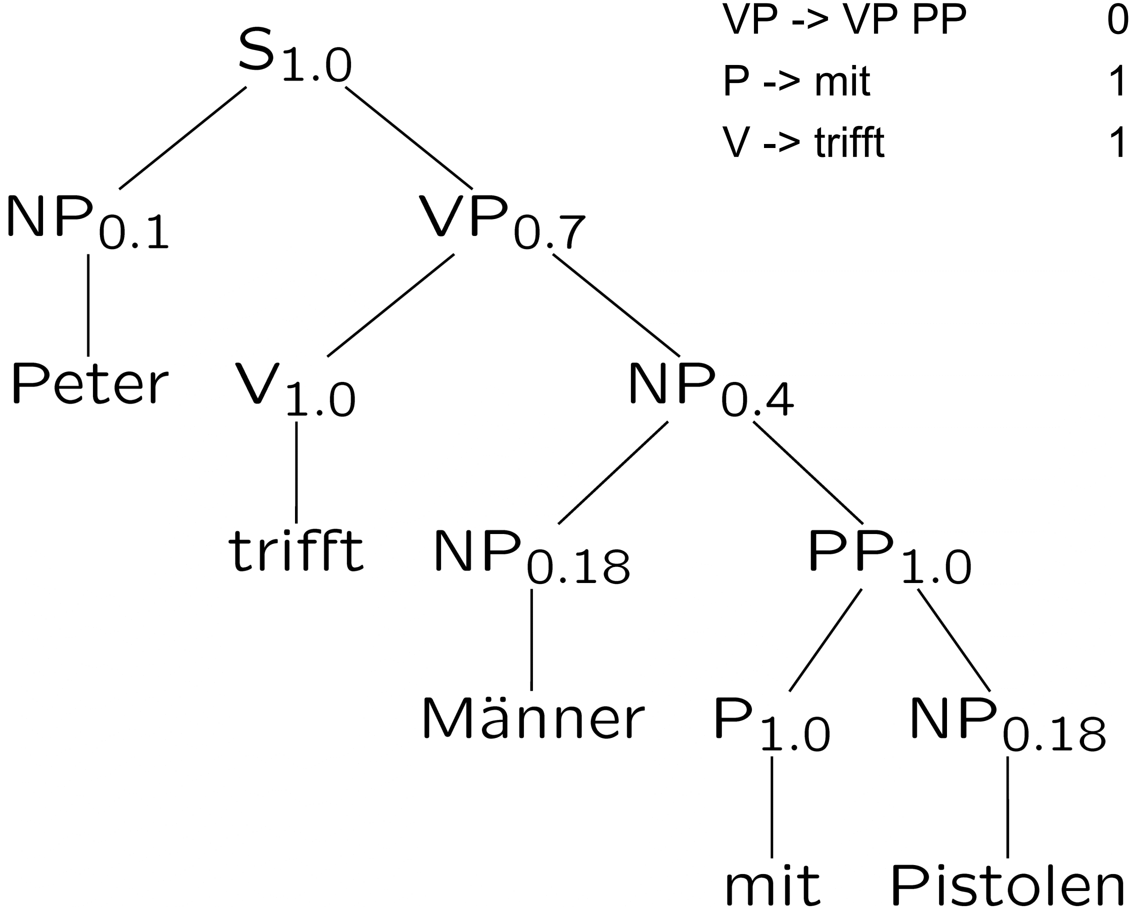
Beispiel

S -> NP VP	1.0	NP -> NP PP	0.4
PP -> P NP	1.0	NP -> Peter	0.1
VP -> V NP	0.7	NP -> Pistolen	0.18
VP -> VP PP	0.3	NP -> Sägen	0.04
P -> mit	1.0	NP -> Männer	0.18
V -> trifft	1.0	NP -> Teleskope	0.1

Satz: „Peter trifft Männer mit Pistolen."

Analyse 1

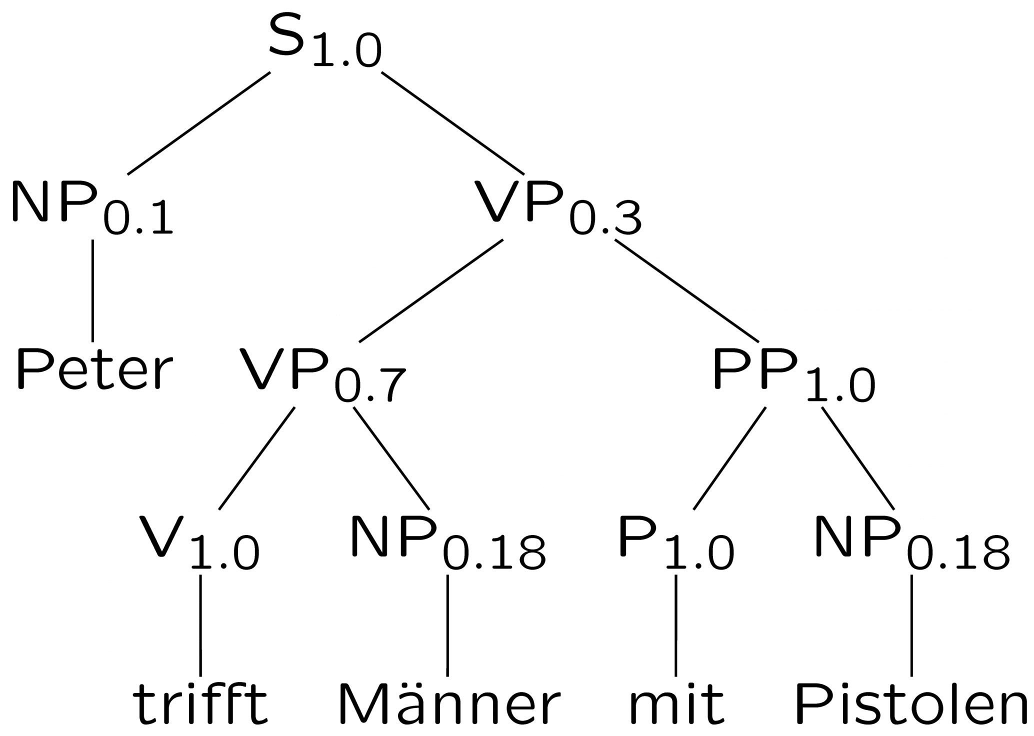
S -> NP VP	1.0	NP -> NP PP	0.4
PP -> P NP	1.0	NP -> Peter	0.1
VP -> V NP	0.7	NP -> Pistolen	0.18
VP -> VP PP	0.3	NP -> Sägen	0.04
P -> mit	1.0	NP -> Männer	0.18
V -> trifft	1.0	NP -> Teleskope	0.1



$$P(t) = 1.0 * 0.1 * 0.7 * 1.0 * 0.4 * 0.18 * 1.0 * 1.0 * 0.18 = 0.0009072$$

Analyse 2

S -> NP VP	1.0	NP -> NP PP	0.4
PP -> P NP	1.0	NP -> Peter	0.1
VP -> V NP	0.7	NP -> Pistolen	0.18
VP -> VP PP	0.3	NP -> Sägen	0.04
P -> mit	1.0	NP -> Männer	0.18
	1.0	NP -> Teleskope	0.1



$$P(t) = 1.0 * 0.1 * 0.3 * 0.7 * 1.0 * 0.18 * 1.0 * 1.0 * 0.18 = 0.0006804$$

Zwischenstand

- PCFG (probab. kontextfreie Grammatik)
  - Regeln der Form X -> A B ...
  - P(Regel) = P(RS | LS) = P(Expansion | Nonterminal)
  - P(Baum) = Produkt aller P(Regeln)
- Vorteile
  - PCFG kann aus Baumbank abgelesen werden
  - P(Baum) kann als Maß für Grammatikalität interpretiert werden
    - Gute Analysen: hohe Wahrscheinlichkeit
    - Schlechte Analysen: niedrige Wahrscheinlichkeit

Aber stimmt das wirklich?

Evaluation von Baumbank-PCFGs

- Trainingsteil: Extraktion einer Grammatik
- Testteil: Vergleich des wahrscheinlichsten Baumes der gelernten Grammatik (GB) mit dem Baumbank-Baum (BB)
  - Precision: Anteil der Regeln im GB, die auch im BB vorkommen
  - Recall: Anteil der Regeln im BB, die auch im GB vorkommen
- Englisch (Penn Treebank):

Satzlänge	∅ Länge	Precision	Recall
2-16 Test	11.4	85.0	87.7
2-25 Test	16.3	82.0	84.0
2-40 Test	21.9	78.8	80.4

Ergebnisse (Deutsch)

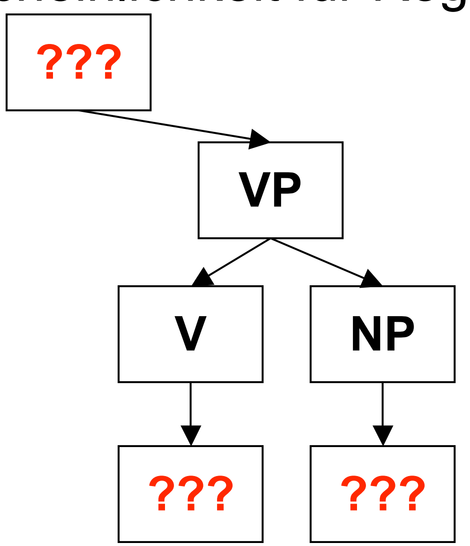
- Baumbank-PCFG für Deutsch (NEGRA)
  - 72.99 Recall, 70.00 Precision

- Wahrscheinlichkeit von Baumbank-PCFGs modelliert Grammatikalität nicht optimal
  - 1 von 5 Regeln des BB (E) fehlt
  - 1 von 5 Regeln des GB (E) ist dazuerfunden

Woher kommen die Fehler?

Produkte von Regelwahrscheinlichkeiten

- Was bedeutet es, die P(R) zu multiplizieren?
  - Jede Regel sieht sich selbst (Unabhängigkeitsannahme)
    - Ob R1 vorkommt, hat keinen Einfluss auf R2
    - Wahrscheinlichkeit für Regel nur vom lokalem Baum abhängig

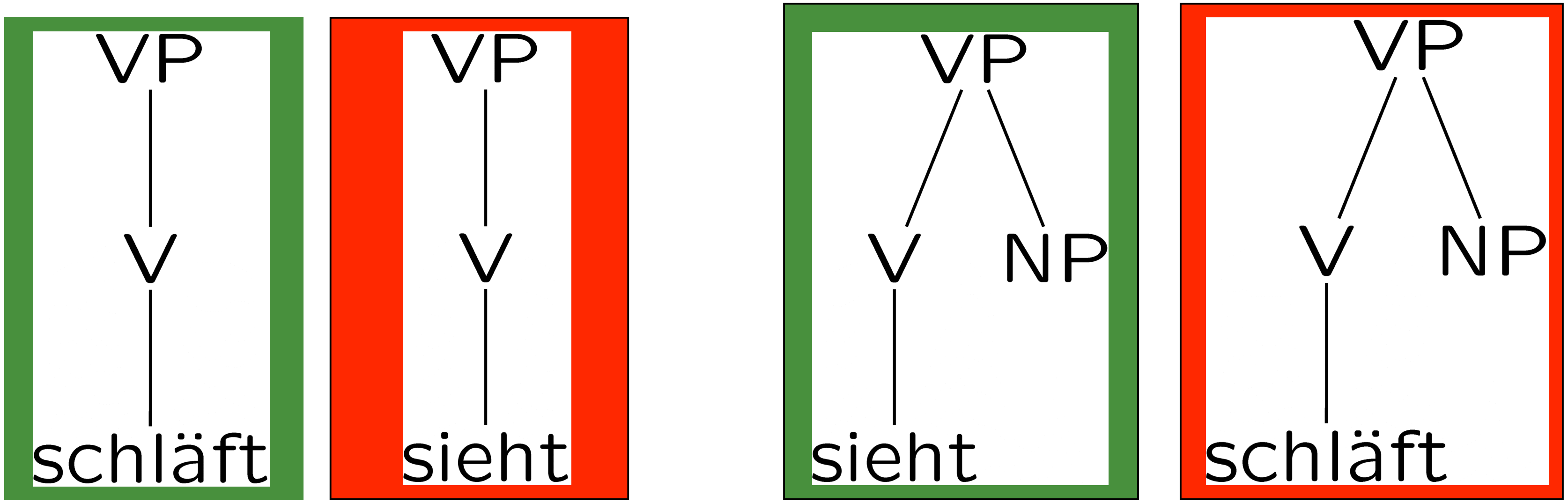


P(V NP | VP) = wie oft expandiert VP zu V NP?

Linguistisch falsche Annahme!

Problem 1: Einfluss der Kinder bei VPs

- Transitive Verben: Hohes P für VP -> V **NP**
- Intransitive Verben: Hohes P für VP -> V



Nicht mit Regelwahrscheinlichkeiten
P(V | VP) und P(V NP | VP) ausdrückbar!

Lexikalisierung

- Regel um lexikalischen Kopf erweitern
  - Hängt von Phrasentyp ab
    - VPs, Ss sehen ihr (Haupt-) verb
    - NPs sehen ihr Nomen
    - APs sehen ihr Adjektiv

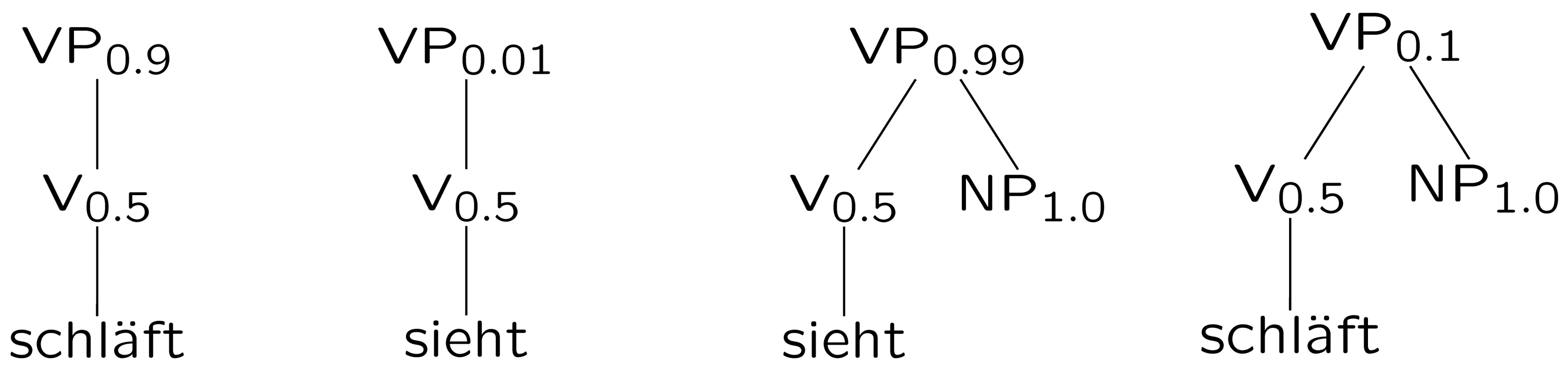
- Form: $P(V \mid VP, schlafen) > P(V\ NP \mid VP, schlafen)$
  - „Die Wahrscheinlichkeit, dass eine VP, deren Kopf „schlafen“ ist, nur aus einem V besteht (intransitiv ist), ist größer als die Wahrscheinlichkeit, dass sie aus einem V und einer NP besteht“

Lexikalisierung (Beispiel)

- P(V NP | VP, schlafen) = 0.1
- P(V | VP, schlafen) = 0.9
- P(V NP | VP, sehen) = 0.99
- P(V | VP, sehen) = 0.01

P(schlafen|V) = 0.5
P(sehen|V) = 0.5

- P(NP) = 1.0



P(t) = 0.45   P(t)=0.005   P(t) = 0.495   P(t) = 0.05

Korrektheit und Grammatikgröße

- Lexikalisierung modelliert linguistischen Zusammenhang
  - Erhöht im Prinzip **Korrektheit**
    - Sinnvolle Parsebäume erhalten höhere Wahrscheinlichkeit

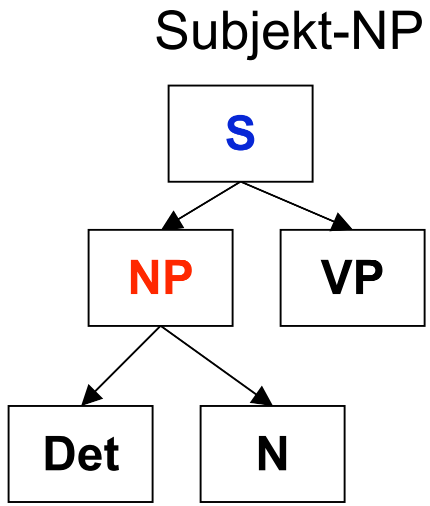
Korrektheit und Grammatikgröße

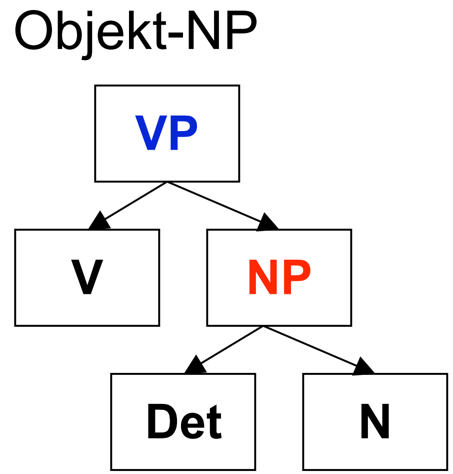
- Aber: Modell wird sehr groß
  - Je eine Regel für Kombination (Regel,Kopfwort)
    - An Stelle von P(V | VP) haben wir
      - P(V | VP, abfahren),
      - P(V | VP, anrufen),
      - P(V | VP, angeln),
      - P(V | VP, ausrufen)
      - Etc.
  - Resultat: Sehr großer Ereignisraum
    - **Sparse Data-Problem**
      - Gesehene Frequenzen verteilen sich auf mehr Ereignisse (bis zu 1000mal mehr Ereignisse)
      - Was tun für ungesehene Verben?

Konkrete Modelle können auch schlechter werden

Problem 2: Einfluß der Eltern bei NPs

- Regeln wissen nicht, was „über ihnen" passiert
  - Sogenannte Großvaterkategorie





- NP-Regel NP -> Det N ist identisch
  - P(Det N | NP) kann nicht zwischen Subj und Obj unterscheiden

Eltern von NPs

- Wahrscheinlichkeiten für Regeln unterscheiden sich aber in Wirklichkeit zwischen Subj und Obj:

Regel	% als Subjekt	% als Objekt
NP -> PRP	13.7%	2.1%
NP -> NP PP	5.6%	14.1%
NP -> NP S	0.5%	2.6%
NP -> DET N	5.6%	4.6%

- **Lösung**: Knoten kennen ihre Großvaterkategorie (Regeln mit Geschichte / History)
  - P(Det N | NP,**S**) = „NP -> Det N als Subjekt“
  - P(Det N | NP,**VP**) = „NP -> Det N als Objekt“

Beispiel für Regeln mit Geschichte

Regel	% als Subjekt	% als Objekt
NP -> PRP	13.7%	2.1%
NP -> NP PP	5.6%	14.1%
NP -> NP S	0.5%	2.6%
NP -> DET N	5.6%	4.6%

P(PRP | NP,S) = 0.137
P(PRP | NP,VP) = 0.021

P(NP S | NP,S) = 0.005
P(NP S | NP,VP) = 0.026

P(NP PP | NP, S) = 0.056
P(NP PP | NP, VP) = 0.141

P(DET N | NP,S) = 0.056
P(DET N | NP,VP) = 0.046

Regeln mit Geschichte

- Geschichte modelliert wichtigen linguistischen Zusammenhang
  - Hilft auch bei anderen Unterscheidungen:
    - Akkusativ- und Dativ-Objekt (Pronominalisierung)
    - Hauptsatz und Nebensatz
      (D: Verbzweitsatz vs. Verbletztsatz)
  - Höhere Wahrscheinlichkeiten für korrekte Konstruktionen

- Wieder mehr Regeln
  - P(RS | LS, GV): Kombination aus Regel und Großvaterkategorie
  - Anzahl der Großvaterkategorien beschränkt
    - Nicht so schlimm wie bei Lexikalisierung
    - Beispiel NP: Regelzahl verdoppelt sich (Subjekt, Objekt)
  - Sparse data-Problem existiert, aber weniger dramatisch

Evaluation: State of the art

- Parser verwenden Geschichte und Lexikalisierung
  - Backoff-Verfahren:
    - Wenn lexikalischer Kopf unbekannt, verzichte auf Lex.
    - Wenn Grossvaterkategorie unbekannt, verzichte auf Geschichte

- Englisch (Penn Treebank): Recall 90%, Precision 90%
  - (Naives Modell Recall 79%, Precision 80%)

- Deutsch (NEGRA): Recall 81%, Precision 77%
  - (Naives Modell Recall 73%, Precision 70%)

Bewertung

- Wie gut sind die Parser wirklich?
  - Typische Satzlänge: 20 Wörter
    - Typische Baumgröße: 20 Regeln
    - Chance für vollständig richtigen Baum:
      - Bei 90% Akkuratheit (Englisch) grob etwa 12%
      - Bei 80% Akkuratheit (Deutsch) grob etwa 1%
- Warum ist Deutsch schwieriger?
  - Flexible Wortstellung
  - Flachere Annotation der Bankbanken

Parsing mit PCFGs

1. Lerne Regeln mit Wahrscheinlichkeiten
2. Berechne Wahrscheinlichkeiten für Bäume aus Regelwahrscheinlichkeiten
3. Finde effizientes Verfahren, um wahrscheinlichste Bäume zu konstruieren

Herkömmliches Parsing

- Top-Down: Fange an mit S, und wende alle anwendbaren Regeln an
  - Beispiel: Earley-Algorithmus
- Bottom-Up: Fange an mit Worten, kombiniere mit allen anwendbaren Regeln

Erschöpfendes (exhaustives) Parsing

Parsing mit Baumbank-Grammatiken

- Viele Regeln
  - Exhaustives Parsing nicht beherrschbar für Sätze mit mehr als 10 Wörtern

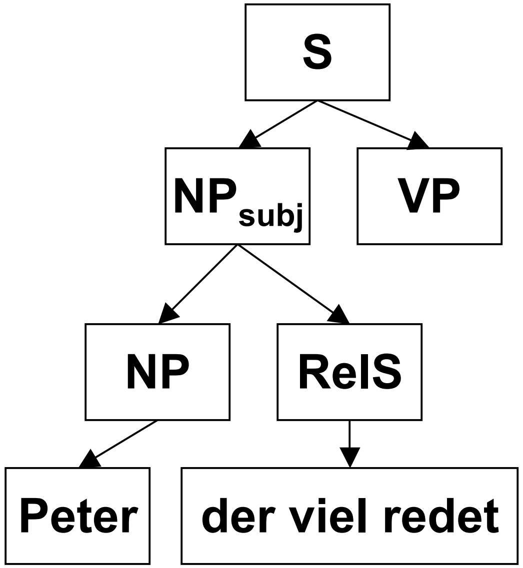
- Hoffnung: ungleiche Wahrscheinlichkeitsverteilung
  - Es gibt wenig „vernünftige“ Parsebäume, viel Müll
  - „Vernünftige“ Parsebäume sind viel wahrscheinlicher als (fast) alle anderen

Aufgabe: Konstruiere die kleine Anzahl sehr wahrscheinlicher Parsebäume

Determinisches Parsing

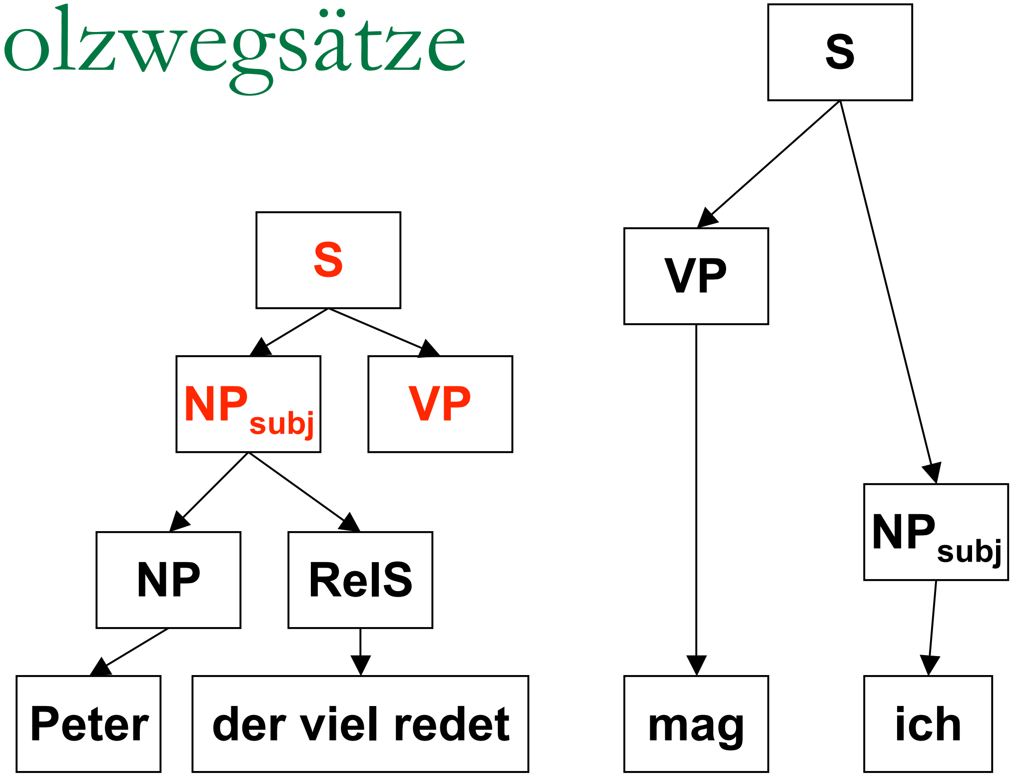
- Extremansatz: verfolge eine einzige Hypothese
  - Beginne irgendwo
  - Finde den lokal wahrscheinlichsten Parsebaum
  - Finde wahrscheinlichste Erweiterung
  - Wiederhole
- Sehr effizient (lineare Laufzeit)
- Erhält man so immer den global wahrscheinlichsten Parsebaum? Nein!

Holzwegsätze



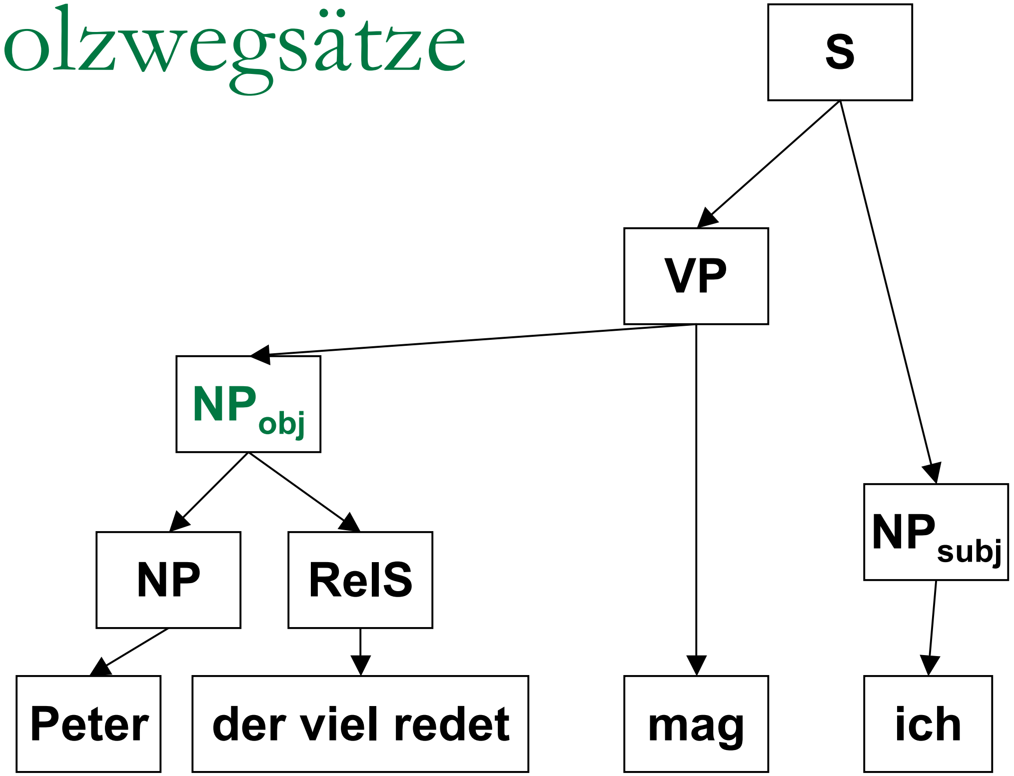
Wahrscheinlichste Analyse einer NP:
Als Subjekt eines Satzes (VP fehlt)

Holzwegsätze



Verb + echtes Subjekt machen Subjektanalyse unmöglich: "ich" kann nur Subjekt sein

Holzwegsätze



Löschung von VP und S
Reanalyse der NP als Objekt

Parsing und lokale Ambiguitäten

- Holzwegsätze sind lokale Ambiguitäten:
  - Ein Teil des Satzes hat (mind.) zwei Analysen
  - Wahrscheinlichere Analyse läßt sich nicht zu Analyse des gesamten Satzes ergänzen
    - Deterministisches Parsing folgt nur der wahrscheinlicheren Analyse
    - Parsing schlägt fehl: Satz erhält überhaupt keine Analyse
- Stattdessen: Beam Search

Beam Search

- Verfolge die n besten Analysen
  - Verallgemeinerung von deterministischem Parsing
- Datenstruktur: Liste von halbfertigen Parsebäumen (Hypothesen) mit Wahrscheinlichkeiten
  - Jede Hypothese um jede anwendbare Regel erweitern
  - Wahrscheinlichkeiten für erweiterte Hypothesen ausrechnen
  - Die n besten Hypothesen behalten, die anderen vergessen
  - Wiederholen, bis n Hypothesen für ganzen Satz

Beam Search

- $n = 1$: deterministisches Parsing
- $n = \infty$: exhaustives Parsing

- Gefahr: für kleines n kann der global beste Parsebaum „aus dem Strahl fallen“
  - Wie groß muß n sein?

Beam Search beim Parsen

n	Zeit (s)	Precision	Recall
20	2.07	87.9	87.1
15	1.58	87.7	86.9
10	1.07	87.7	86.9
7	0.76	87.4	86.6
5	0.56	87.3	86.8
2	0.35	85.1	86.1
1	0.14	82.4	83.4

Beam Search in der Praxis

- Optimales n muß empirisch ermittelt werden
- Funktioniert ziemlich gut
  - Bester Parsebaum (fast) immer in der vorne dabei
    - Bestätigt Hypothese über ungleiche Verteilung
    - Grammatische Sätze erhalten hohe Wahrscheinlichkeit
- Beam Search nur Methode, um Suchraum einzuschränken („Filter“)
  - Läßt offen, wie Hypothesen konstruiert werden
    - Top-down parsing: Beginne mit S
    - Bottom-up parsing: Beginne mit Worten

Zusammenfassung

- Kontextfreie Grammatiken
  - Können aus Baumbanken gelernt werden
  - Viele Regeln
  - Hohe Ambiguität

- Probabilistische kontextfreie Grammatiken
  - Möglichkeit, über „besten Baum“ zu sprechen
  - Regelwahrscheinlichkeiten ablesen
    - Einfache Regeln: kein gute Modellierung von Grammatikalität
    - Erweiterung mit Lexikalisierung und Geschichte
  - Suche nach besten Analysen mit Beam Search