Parsing Grundlagen und Chart Parsing

Katja Markert

Institut für Computerlinguistik
Uni Heidelberg
markert@cl.uni-heidelberg.de
mit einigen Folien von Yannick Versley

December 12, 2019

Übersicht

Übersicht

Parsing

Parsing

Gegeben eine CFG. Wie kann für einen Satz *s* automatisch

1. das Erkennungsproblem gelöst werden?
2. die syntaktische Analyse durchgeführt werden?

Eingabe: Ein Satz *s* und eine CFG

Ausgabe: Ein oder mehrere Phrasenstrukturbäume, falls *s* grammatisch. *False* sonst.

Warum Parsing?

- Grammar Checking
- Zwischenstufe für tiefe semantische Verarbeitung (Informationsextraktion, Question-Answering)
  - [$_S$ [$_{NP}$ Who] [$_{VP}$ won [$_{NP}$ the Nobel Prize for Physics] in 2007] ?]
  - [$_S$ [$_{NP}$ France's Albert Fert and Germany's Peter Gruenberg] [$_{VP}$ win [$_{NP}$ the 2007 Nobel Prize for Physics]] it was announced Tuesday Oct 9 2007 .]

Übersicht

Parsingstrategien

Zwei Informationsquellen beschränken die Suche:

1. Eingabesatz: Jeder Parsebaum muss die Wörter des Eingabesatzes als Blätter (Terminalsymbole) enthalten
2. Startsymbol der Grammatik: Jeder Parsebaum muss Startsymbol als Wurzelknoten besitzen

Daraus ergeben sich zwei Grundlegende Parsingstrategien: Bottom-Up (datengetrieben) und Top-Down (zielgetrieben)

Top-Down Parsing

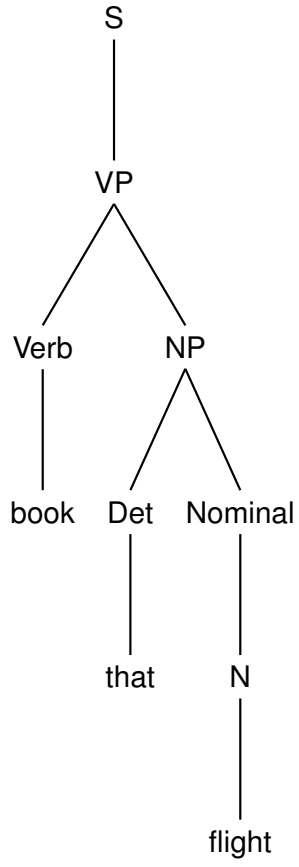
Expandiert Grammatikregeln ausgehend vom Startsymbol. Die Suche ist erfolgreich, wenn die Blätter des Parsebaums genau mit den Wörtern des Eingabestrings übereinstimmen.

Recursive Descent Algorithmus

- top-down
- left-right
- erschöpfende depth-first Suche mit Backtracking
- Regeln in Grammatikreihenfolge

Beispiel-CFG L_1,Jurafsky und Martin, Kap 13, 3rd ed.

$S \rightarrow NP\ VP$	$Det \rightarrow that \mid this \mid a$
$S \rightarrow Aux\ NP\ VP$	$Noun \rightarrow book \mid flight \mid meal \mid money$
$S \rightarrow VP$	$Verb \rightarrow book \mid include \mid prefer$
$NP \rightarrow Pronoun$	$Pronoun \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$Proper\text{-}Noun \rightarrow Houston \mid TWA$
$NP \rightarrow Det\ Nominal$	$Aux \rightarrow does$
$Nominal \rightarrow Noun$	$Preposition \rightarrow from \mid to \mid on \mid near \mid through$
$Nominal \rightarrow Nominal\ Noun$	
$Nominal \rightarrow Nominal\ PP$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb\ NP$	
$VP \rightarrow Verb\ NP\ PP$	
$VP \rightarrow Verb\ PP$	
$VP \rightarrow VP\ PP$	
$PP \rightarrow Preposition\ NP$	



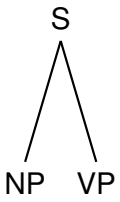
Beispielsatz S_1: book that flight

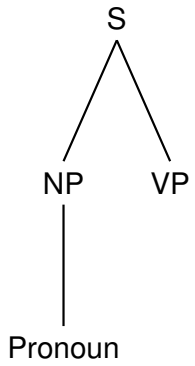
Beispiel Recursive Descent für *book that flight*

S

$S \rightarrow NP\ VP$	$Det \rightarrow that \mid this \mid a$
$S \rightarrow Aux\ NP\ VP$	$Noun \rightarrow book \mid flight \mid meal \mid money$
$S \rightarrow VP$	$Verb \rightarrow book \mid include \mid prefer$
$NP \rightarrow Pronoun$	$Pronoun \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$Proper\text{-}Noun \rightarrow Houston \mid TWA$
$NP \rightarrow Det\ Nominal$	$Aux \rightarrow does$
$Nominal \rightarrow Noun$	$Preposition \rightarrow from \mid to \mid on \mid near \mid through$
$Nominal \rightarrow Nominal\ Noun$	
$Nominal \rightarrow Nominal\ PP$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb\ NP$	
$VP \rightarrow Verb\ NP\ PP$	
$VP \rightarrow Verb\ PP$	
$VP \rightarrow VP\ PP$	
$PP \rightarrow Preposition\ NP$	

Beispiel Recursive Descent für *book that flight*

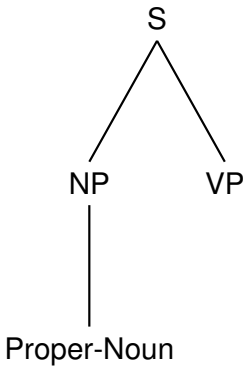




Backtracking nötig, da POS Mismatch

$S \rightarrow NP\ VP$	$Det \rightarrow that \mid this \mid a$
$S \rightarrow Aux\ NP\ VP$	$Noun \rightarrow book \mid flight \mid meal \mid money$
$S \rightarrow VP$	$Verb \rightarrow book \mid include \mid prefer$
$NP \rightarrow Pronoun$	$Pronoun \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$Proper\text{-}Noun \rightarrow Houston \mid TWA$
$NP \rightarrow Det\ Nominal$	$Aux \rightarrow does$
$Nominal \rightarrow Noun$	$Preposition \rightarrow from \mid to \mid on \mid near \mid through$
$Nominal \rightarrow Nominal\ Noun$	
$Nominal \rightarrow Nominal\ PP$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb\ NP$	
$VP \rightarrow Verb\ NP\ PP$	
$VP \rightarrow Verb\ PP$	
$VP \rightarrow VP\ PP$	
$PP \rightarrow Preposition\ NP$	

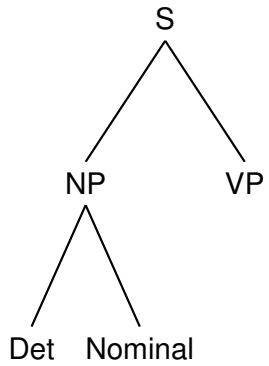
Beispiel Recursive Descent für *book that flight*



Backtracking nötig, da POS Mismatch

$S \rightarrow NP\ VP$	$Det \rightarrow that \mid this \mid a$
$S \rightarrow Aux\ NP\ VP$	$Noun \rightarrow book \mid flight \mid meal \mid money$
$S \rightarrow VP$	$Verb \rightarrow book \mid include \mid prefer$
$NP \rightarrow Pronoun$	$Pronoun \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$Proper\text{-}Noun \rightarrow Houston \mid TWA$
$NP \rightarrow Det\ Nominal$	$Aux \rightarrow does$
$Nominal \rightarrow Noun$	$Preposition \rightarrow from \mid to \mid on \mid near \mid through$
$Nominal \rightarrow Nominal\ Noun$	
$Nominal \rightarrow Nominal\ PP$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb\ NP$	
$VP \rightarrow Verb\ NP\ PP$	
$VP \rightarrow Verb\ PP$	
$VP \rightarrow VP\ PP$	
$PP \rightarrow Preposition\ NP$	

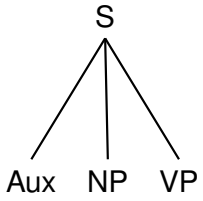
Beispiel Recursive Descent für *book that flight*



Backtracking nötig, da POS Mismatch

$S \rightarrow NP\ VP$	$Det \rightarrow that \mid this \mid a$
$S \rightarrow Aux\ NP\ VP$	$Noun \rightarrow book \mid flight \mid meal \mid money$
$S \rightarrow VP$	$Verb \rightarrow book \mid include \mid prefer$
$NP \rightarrow Pronoun$	$Pronoun \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$Proper\text{-}Noun \rightarrow Houston \mid TWA$
$NP \rightarrow Det\ Nominal$	$Aux \rightarrow does$
$Nominal \rightarrow Noun$	$Preposition \rightarrow from \mid to \mid on \mid near \mid through$
$Nominal \rightarrow Nominal\ Noun$	
$Nominal \rightarrow Nominal\ PP$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb\ NP$	
$VP \rightarrow Verb\ NP\ PP$	
$VP \rightarrow Verb\ PP$	
$VP \rightarrow VP\ PP$	
$PP \rightarrow Preposition\ NP$	

Beispiel Recursive Descent für *book that flight*



Backtracking nötig, da POS Mismatch

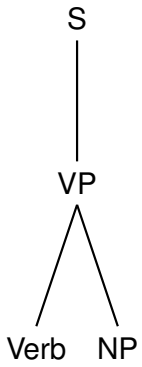
$S \rightarrow NP\ VP$	$Det \rightarrow that \mid this \mid a$
$S \rightarrow Aux\ NP\ VP$	$Noun \rightarrow book \mid flight \mid meal \mid money$
$S \rightarrow VP$	$Verb \rightarrow book \mid include \mid prefer$
$NP \rightarrow Pronoun$	$Pronoun \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$Proper\text{-}Noun \rightarrow Houston \mid TWA$
$NP \rightarrow Det\ Nominal$	$Aux \rightarrow does$
$Nominal \rightarrow Noun$	$Preposition \rightarrow from \mid to \mid on \mid near \mid through$
$Nominal \rightarrow Nominal\ Noun$	
$Nominal \rightarrow Nominal\ PP$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb\ NP$	
$VP \rightarrow Verb\ NP\ PP$	
$VP \rightarrow Verb\ PP$	
$VP \rightarrow VP\ PP$	
$PP \rightarrow Preposition\ NP$	

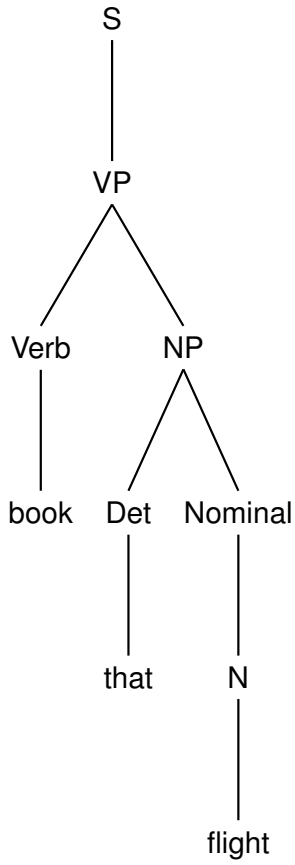
Beispiel Recursive Descent für *book that flight*

Backtracking nötig, das zuviel Input übrig

$S \rightarrow NP\ VP$	$Det \rightarrow that \mid this \mid a$
$S \rightarrow Aux\ NP\ VP$	$Noun \rightarrow book \mid flight \mid meal \mid money$
$S \rightarrow VP$	$Verb \rightarrow book \mid include \mid prefer$
$NP \rightarrow Pronoun$	$Pronoun \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$Proper\text{-}Noun \rightarrow Houston \mid TWA$
$NP \rightarrow Det\ Nominal$	$Aux \rightarrow does$
$Nominal \rightarrow Noun$	$Preposition \rightarrow from \mid to \mid on \mid near \mid through$
$Nominal \rightarrow Nominal\ Noun$	
$Nominal \rightarrow Nominal\ PP$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb\ NP$	
$VP \rightarrow Verb\ NP\ PP$	
$VP \rightarrow Verb\ PP$	
$VP \rightarrow VP\ PP$	
$PP \rightarrow Preposition\ NP$	

Beispiel Recursive Descent für *book that flight*





Endlich (nach zwei weiteren Backtrackings an NP)

Top-Down Parsing Terminierung

- Die Blätter des generierten Baumes werden mit den Wörtern des Eingabesatzes abgeglichen. Ist das aktuelle Eingabewort kompatibel mit dem POS, so ist die verwendete Regel erfolgreich
- Können die Blätter des konstruierten Parsebaums mit den Wörtern des Eingabestrings vollständig abgeglichen werden, so ist der konstruierte Baum ein valider Parsebaum.

Recursive Descent Probleme: Effizienz

- Fehlschläge erst bei Abgleich mit Blättern sichtbar
- Probleme mit POS-Tagging Mismatch können durch eine Kombination mit Bottom-Up (left-corner parsing) vermieden werden
- Beim systematischem Rücksetzen müssen identische Teilstrukturen immer wieder berechnet werden (s. nächstes Beispiel)
- Diese Ineffizienz kann durch Bottom-up Kombination nicht vermieden werden

Recursive Descent: Ineffizienz Beispiel

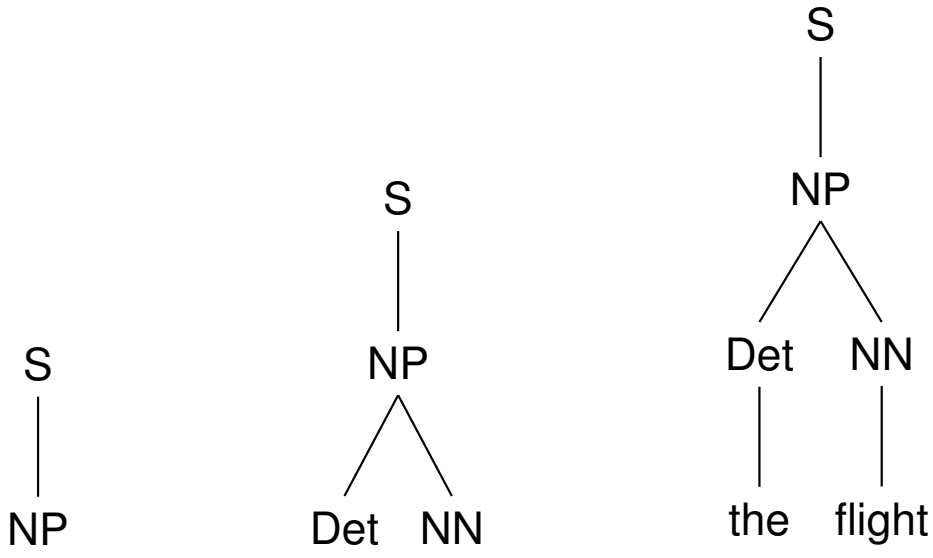
Beispiel für Ineffizienz trotz gutem POS Matching

$$\begin{aligned}
S &\rightarrow NP \\
NP &\rightarrow Det\ NN \\
NP &\rightarrow NP\ PP \\
NP &\rightarrow NN \\
PP &\rightarrow IN\ NP \\
NN &\rightarrow \{\text{flight, London}\} \\
IN &\rightarrow \text{to} \\
Det &\rightarrow \text{the}
\end{aligned}$$

Satz: *the flight to London*

Recursive Descent: Ineffizienz Beispiel

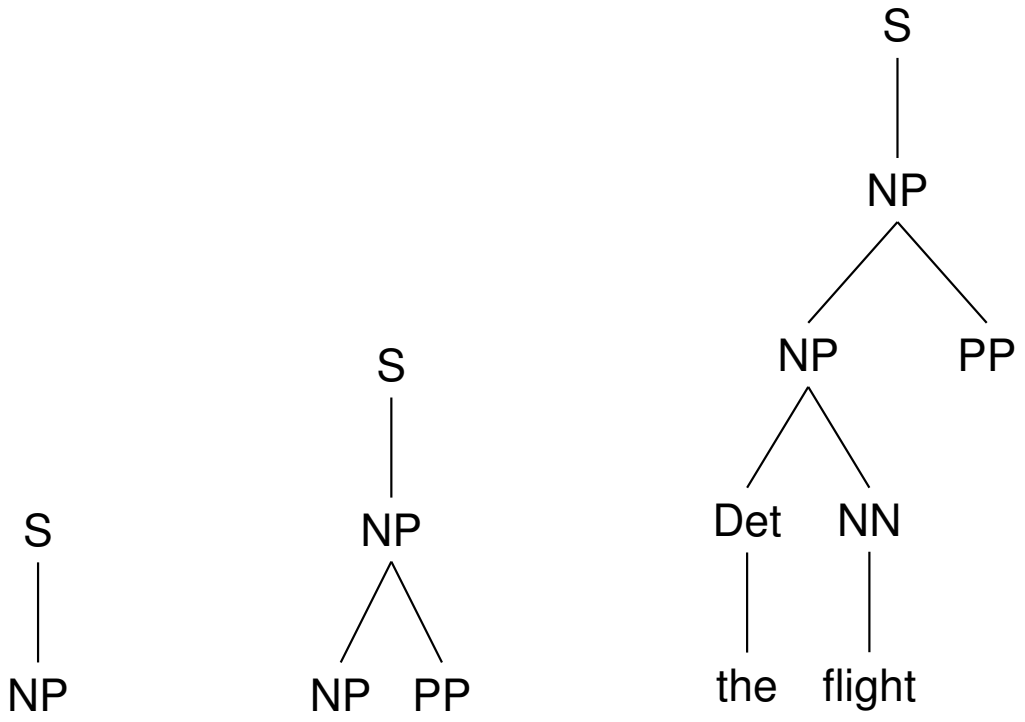
the flight to London



PP kann nicht inkorporiert werden $\rightarrow$ Backtrack

Recursive Descent: Ineffizienz Beispeil

the flight to London



NP *the flight* wurde doppelt abgeleitet. Komplexität im schlimmsten Fall exponentiell zur Eingabelänge

Recursive Descent Probleme: Rekursion

$$S \rightarrow NP$$
$$NP \rightarrow NP\ PP$$
$$NP \rightarrow NNS$$
$$PP \rightarrow IN\ NP$$
$$NNS \rightarrow \{men, hats\}$$
$$IN \rightarrow with$$

Parsen Sie damit Top-Down *men with hats*

Linksrekursion: $A \rightarrow A\ X$. Unendliche Schleife. Kann vermieden werden, wenn man Grammatik anders schreibt.

$$NP \rightarrow NP\ PP$$
$$NP \rightarrow NNS$$

$$NP \rightarrow NNS\ T$$
$$T \rightarrow PP\ T$$
$$T \rightarrow \varepsilon$$

Recursive Descent Probleme: Rekursion

$$S \rightarrow NP$$
$$NP \rightarrow NP\ PP$$
$$NP \rightarrow NNS$$
$$PP \rightarrow IN\ NP$$
$$NNS \rightarrow \{men, hats\}$$
$$IN \rightarrow with$$

Parsen Sie damit Top-Down *men with hats*
Linksrekursion: $A \rightarrow A\ X$. Unendliche Schleife. Kann vermieden werden, wenn man Grammatik anders schreibt.

$$NP \rightarrow NP\ PP$$
$$NP \rightarrow NNS$$

$$NP \rightarrow NNS\ T$$
$$T \rightarrow PP\ T$$
$$T \rightarrow \varepsilon$$

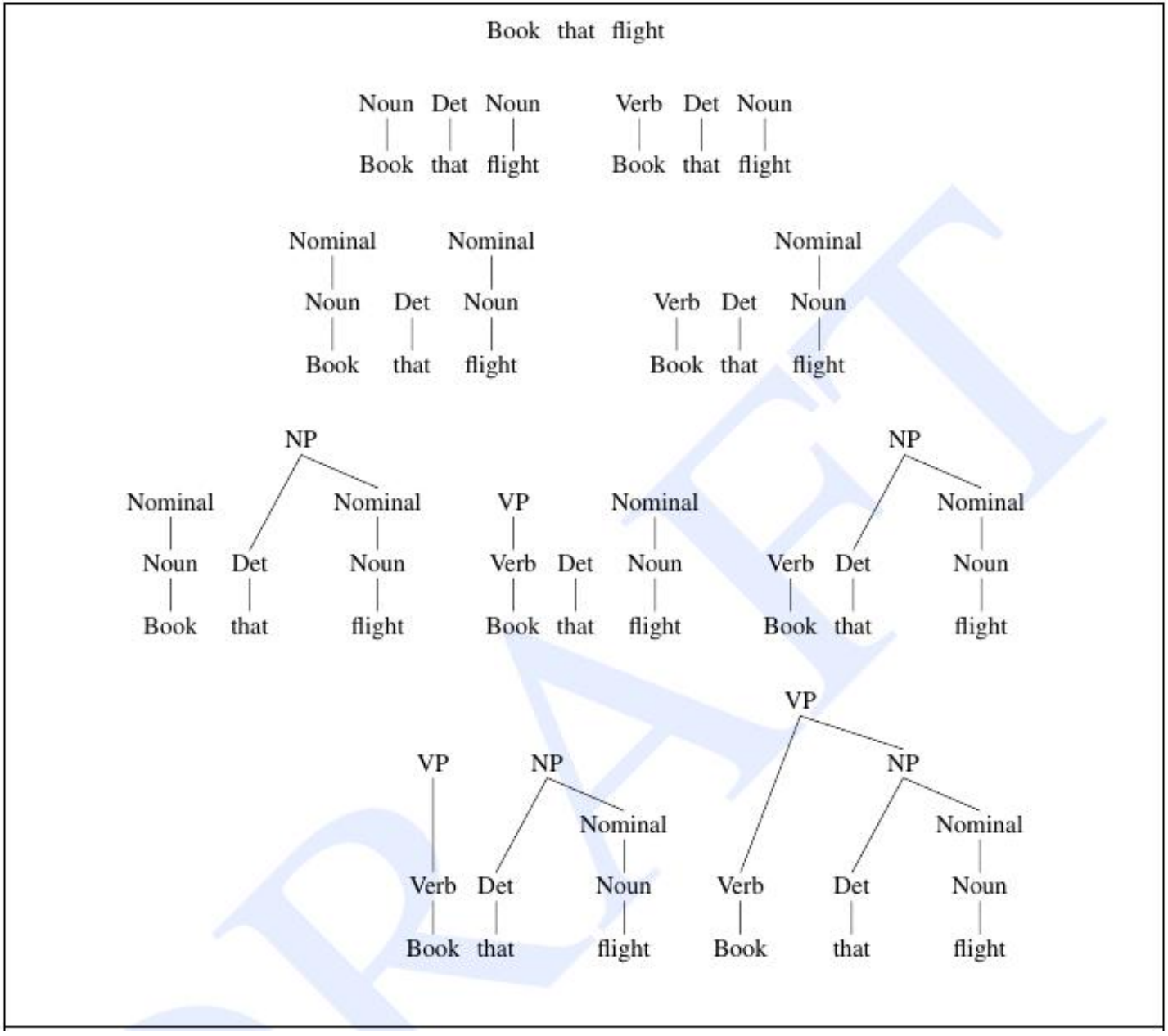
Recursive Descent Probleme: Ambiguität

Wie behandelt der Parser Ambiguität? Lösungen?

Bottom-Up Parsing

- ausgehend von der Eingabesequenz
- generiere Teilbäume auf Basis von Ersetzungsregeln (betrachte rechte Seite), bis das Startsymbol erreicht ist
- erfolgreich, wenn der Parsebaum die gesamte Eingabesequenz überspannt und einen einzigen Wurzelknoten enthält, der mit dem Startsymbol übereinstimmt

Bottom-Up Parsing Beispiel



Vergleich Top-Down und Bottom-Up

Top-Down:

1. exploriert keine ungrammatikalischen Bäume
2. exploriert keine Bäume, die keinen Platz in einem Baum mit Wurzelknoten S finden können
3. generiert viele Bäume, die keinerlei Aussicht haben auf die Eingabewörter zu passen

Bottom-Up:

1. exploriert nur solche Bäume, die mit der Eingabesequenz kompatibel sind
2. generiert viele Bäume, die keinerlei Aussicht haben, in das Startsymbol zu münden

Beide:

1. Viele Teilbäume sind nicht Teil einer vollständigen validen Analyse
2. Viele Teilbäume werden wiederholt berechnet. $\rightarrow$ Dynamische Programmierung/Chart Parsing/CKY

Übersicht

Chart Parsing

- Gesamtparse besteht aus validen Teilbäumen
- Dynamische Programmierung: Speichern von Lösungen für Teilprobleme in Chart zur Vermeidung wiederholter Berechnungen (polynomiale Laufzeit)

Chart

Tabelle, in der Teilresultate der Analyse abgelegt werden. Einträge in Chart heissen Kanten. Eine Kante enthält mindestens:

- Satzabschnitt, auf den sich die Kante bezieht
- Angewandte Syntaxregel

Bottom-Up Chart Parsing (CKY)

- Eingabe: ${}_0\mathit{Book}_1\ \mathit{the}_2\ \mathit{flight}_3\ \mathit{through}_4\ \mathit{Houston}_5$
- Eine Kante der Chart: PP[3,5]. Überspannt die Phrase *through Houston*. Kann sich nur zusammensetzen aus [3,4] und [4, 5]
- CYK Algorithmus: Erstellung von Chart-Kanten (Kasami 1965, Younger 1967, Cocke and Schwary 1970)
- CYK arbeitet nur mit Grammatiken in Chomsky-Normalform.

Unsere Beispielgrammatik L_1

Rechts in Chomsky-Normalform

$S \rightarrow NP\ VP$	$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$	$S \rightarrow X1\ VP$
	$X1 \rightarrow Aux\ NP$
$S \rightarrow VP$	$S \rightarrow book \mid include \mid prefer$
	$S \rightarrow Verb\ NP$
	$S \rightarrow X2\ PP$
	$S \rightarrow Verb\ PP$
	$S \rightarrow VP\ PP$
$NP \rightarrow Pronoun$	$NP \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$	$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$	$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$	$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$	$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$	$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$	$VP \rightarrow Verb\ NP$
$VP \rightarrow Verb\ NP\ PP$	$VP \rightarrow X2\ PP$
	$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$	$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$	$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$	$PP \rightarrow Preposition\ NP$

Figure 13.8 $\mathscr{L}_1$ Grammar and its conversion to CNF. Note that although they aren't shown here all the original lexical entries from $\mathscr{L}_1$ carry over unchanged as well.

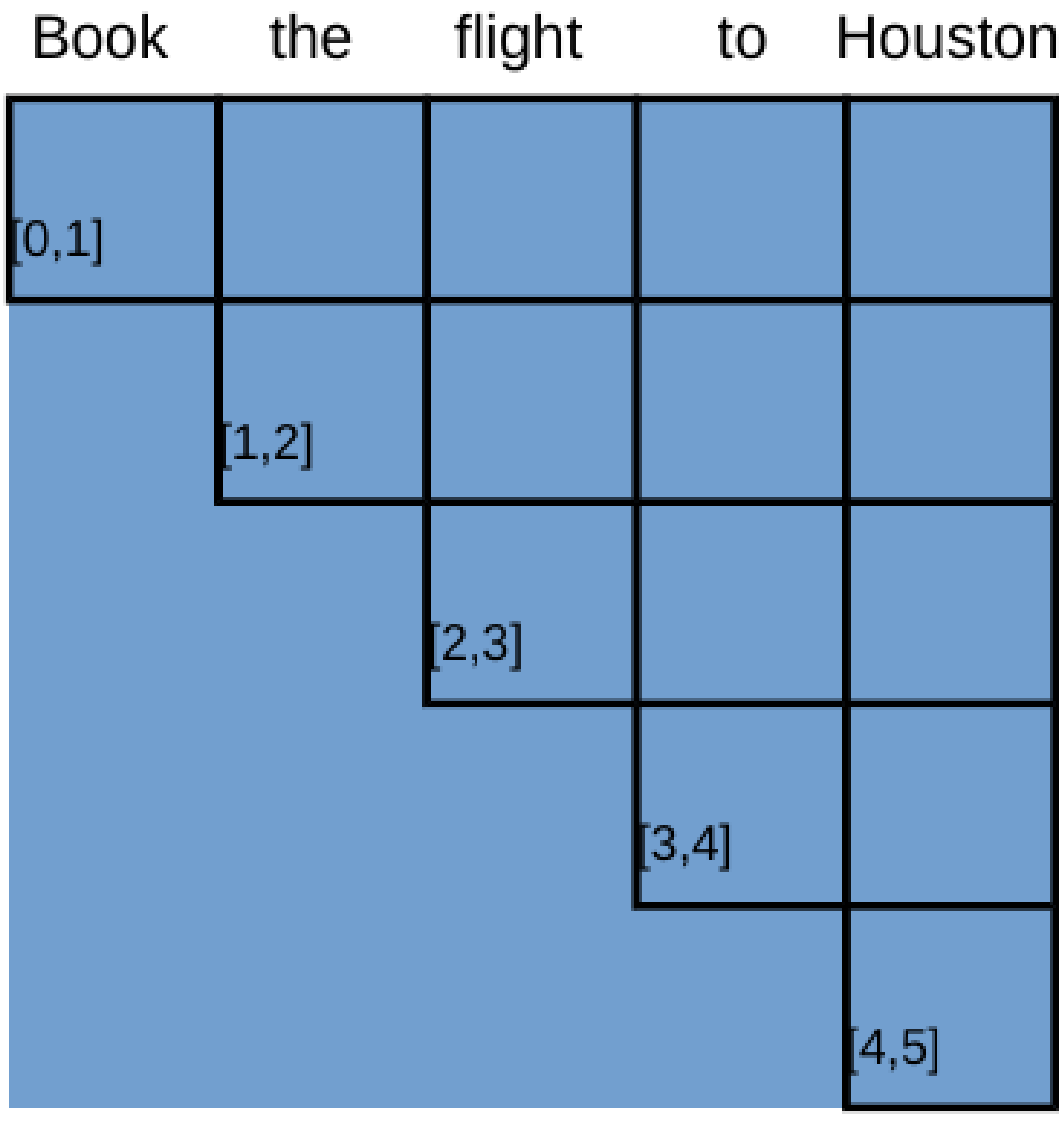
Bild aus Jurafsky und Martin, 3rd edition

Unsere Beispielgrammatik in CNF

S-> NP VP
S→ X1 VP
X1 → Aux NP
S→ Verb NP
S → X2 PP
S→ Verb PP
S→ VP PP
NP→ Det Nominal
Nominal → Nominal Noun
Nominal → Nominal PP
VP→ Verb NP
VP → X2 PP
X2 → Verb NP
VP→ VP PP
PP→ Preposition NP

S→ book |include |prefer
NP → I |she |me
NP → TWA|Houston
Nominal → book | flight | meal | money
VP → book|include|prefer
Verb → book |include|prefer
Noun → book |flight |meal |money
Proper Noun → TWA |Houston
Pronoun →I | she | me
Aux → does
Det → that | this | a | the
Preposition → from | to | on |near |through

CKY Algorithmus am Beispiel



CKY Algorithmus am Beispiel

Initialisierung: Auf Diagonale [j-1,j] stehen mögliche Pos Tags (bottom-up)

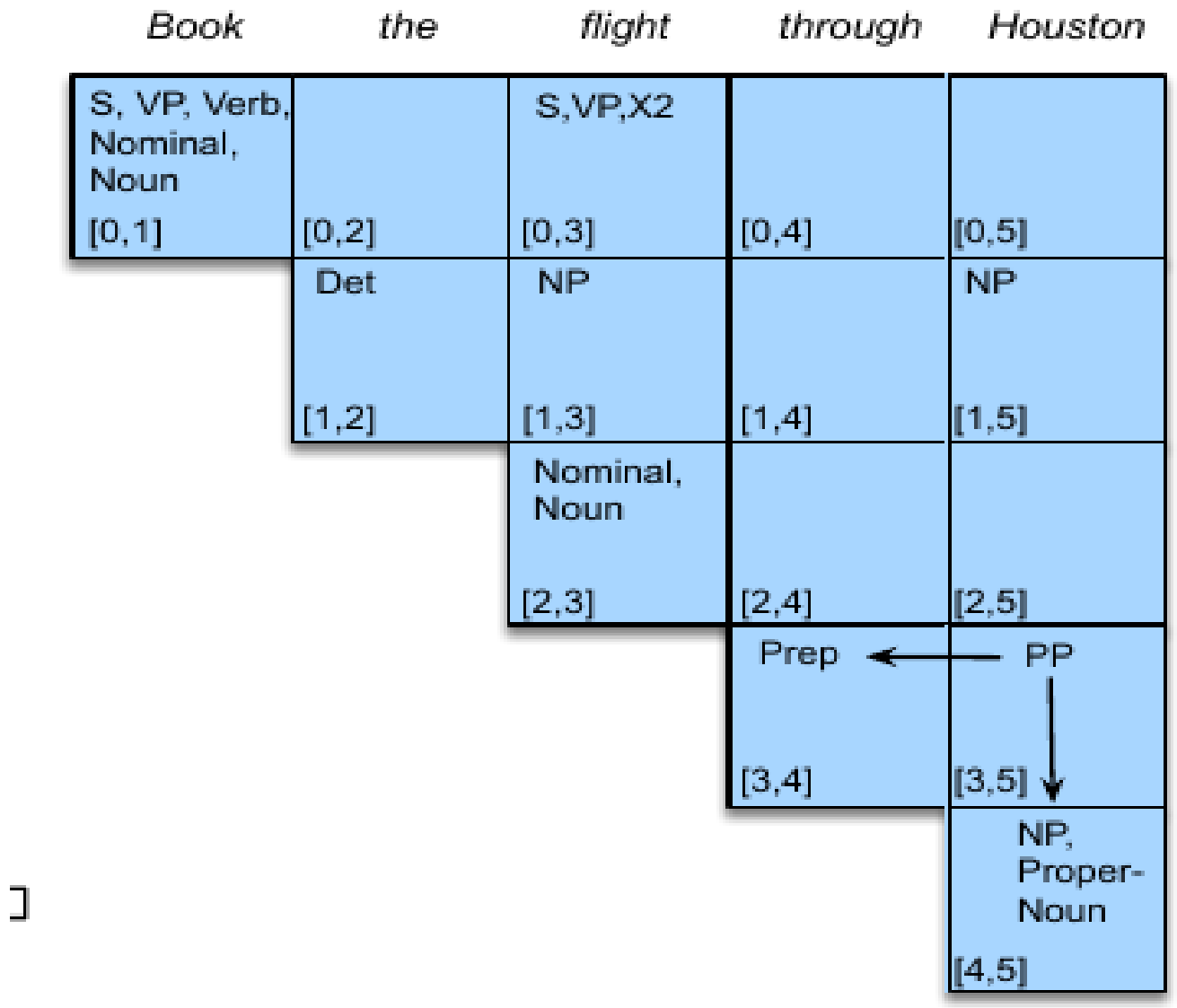
Dann spaltenweise links nach Rechts, unten nach oben

Zelle [i,j] kann sich zusammensetzen aus [i,k] und [k,j] für alle k zwischen i+1 und j-1

Binäre Zusammensetzung → CNF

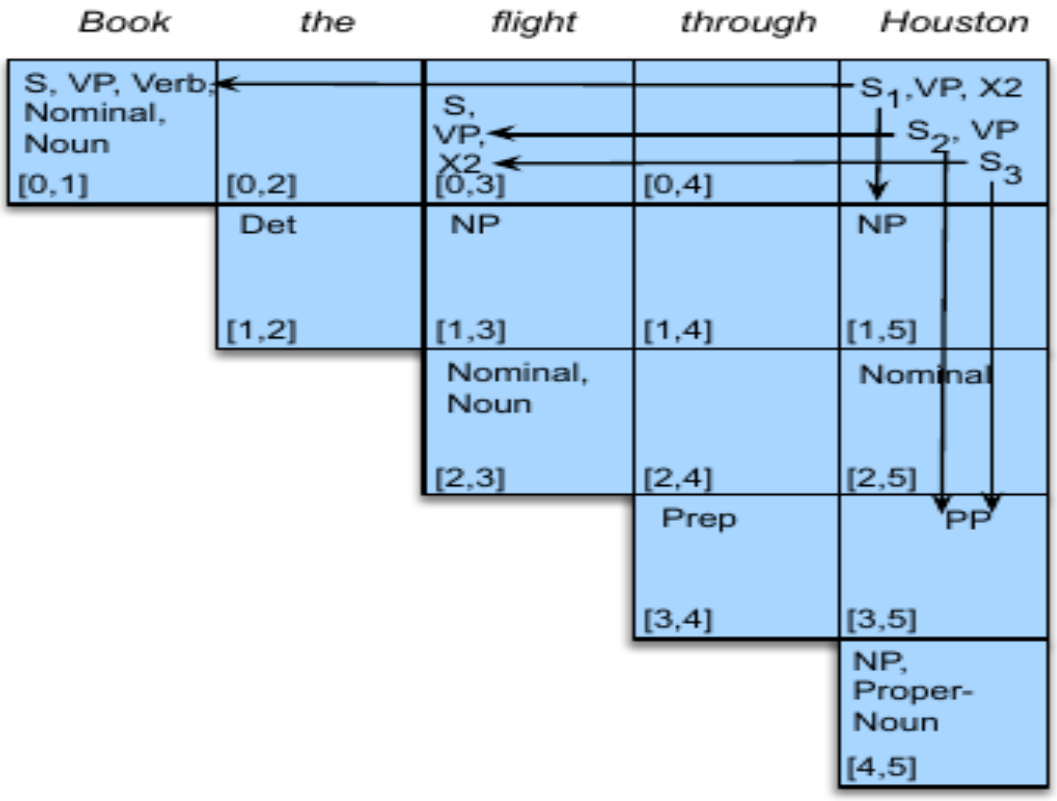
CKY Algorithmus am Beispiel

Am Anfang der letzten Spalte:



CKY Algorithmus am Beispiel

Endergebnis (Bild aus Jurafsky und Martin):



- Obere rechte Ecke: im Buch wurde m.E. ein VP aus X2 PP vergessen
- S_2 und S_3 sind äquivalent und beschreiben beide die Verb Attachment “book through Houston” Lesart.

CKY Algorithmus Pseudocode

```
function CKY-PARSE(words, grammar) returns table

  for j ← from 1 to LENGTH(words) do
    table[j − 1, j] ← {A | A → words[j] ∈ grammar}
    for i ← from j − 2 downto 0 do
      for k ← i + 1 to j − 1 do
        table[i,j] ← table[i,j] ∪
                     {A | A → BC ∈ grammar,
                          B ∈ table[i,k],
                          C ∈ table[k,j]}
```

Figure 13.10 The CKY algorithm

Bild aus Jurafsky und Martin 3rd edition

CKY: Zusammenfassung

- Die Chart berechnet alle möglichen Parsebäume
- Erlaubt das Packen von Ambiguitäten. Chart stellt lokale und globale Ambiguitäten dar.
- Phrasen werden angezeigt (nicht-leere Zellen)
- Laufzeit $O(|R| \cdot n^3)$, wenn n Satzlänge und R die Anzahl der Regeln. Grund: Teilbäume werden nachgeschaut und nicht neu geparst
- Nachteil: braucht CNF

Jurafsky und Martin, 3rd Edition, Kapitel 13.1, 13.2

CKY-Demonstrator `http://lxmls.it.pt/2015/cky.html` (leider mit anderer Chartdarstellung)