Faculty for Informatics

Technical
University
of Munich

Natural Language Processing
IN2361

PD Dr. Georg Groh

Social Computing
Research Group

Chapter 13
Constituency Parsing

- content is based on [1]
- certain elements (e.g. equations or tables) were taken over or taken over in a modified form from [1]
- citations of [1] or from [1] are omitted for legibility
- errors are fully in the responsibility of Georg Groh
- BIG thanks to Dan and James for a great book!

Syntactic Parsing

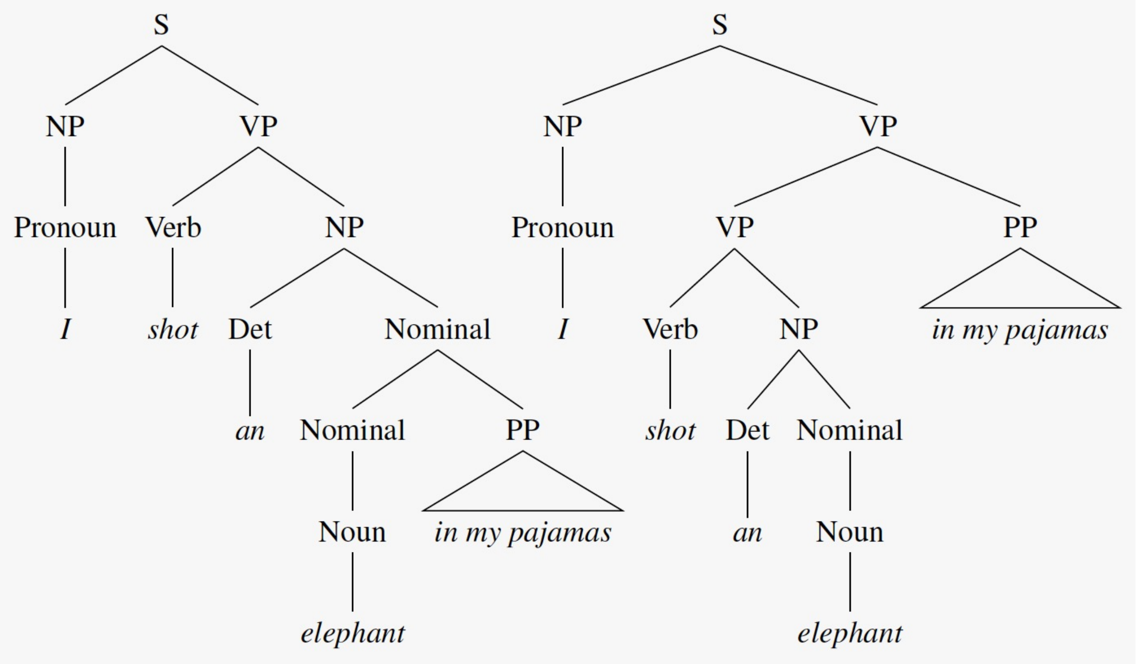
- Syntactic parsing: sentence → parse tree

- applications:
  - grammar checking (e.g. in Word)
  - information extraction + question answering
    What books were written by British women authors before 1800?
  - etc

- classic “conflict”:
  - symbolic reasoning (syntactic + semantic parsing + rules, logic) vs
  - sub-symbolic reasoning (deep learning)

Ambiguity

- (structural) Ambiguity: assign more than one parse tree to a sentence
  - attachment ambiguity: a particular constituent can be attached to the parse tree at more than one place

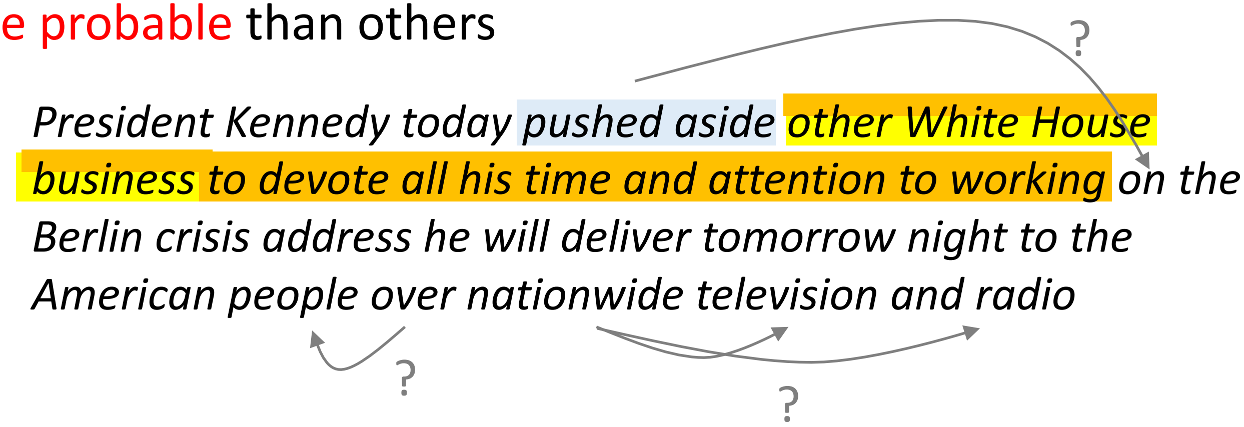


  - coordintion ambiguity: join different sets of phrases by *and*

[old [men and women]]   [old men] and [women]

Disambiguation

- often some parse trees are semantically or statistically or contextually more probable than others



President Kennedy today pushed aside other White House business to devote all his time and attention to working on the Berlin crisis address he will deliver tomorrow night to the American people over nationwide television and radio

→ integrate statistical info into parsers to produce most plausible parse tree (next chapter)

CYK Algorithm

- Cocke Younger Kasami dynamic programming algorithm for parsing CF grammars
    - dynamic programming (as in MinEditDistance, Viterbi, Forward etc. before): systematically fill in tables of solutions to sub-problems, at the end: assemble solution form these
    - two parts: recognizer + actual parser
    - typically: convert to Chomsky Normal Form (CNF) first

CF Grammars: Conversion to CNF

- Convert into Chomsky Normal Form (CNF) : $A \rightarrow B\ C$

  $A \rightarrow w$

1. eliminate terminals from non-pure right hand sides

   $$INF\text{-}VP \rightarrow to\ VP \quad \Longrightarrow \quad \begin{cases} INF\text{-}VP \rightarrow TO\ VP \\ TO \rightarrow to \end{cases}$$

2. eliminate unit productions $A \rightarrow B$

   if $A \overset{*}{\Rightarrow} B$ by a chain of one or more unit productions and $B \rightarrow \gamma$ is a non-unit production in our grammar, then we add $A \rightarrow \gamma$ for each such rule in the grammar and discard all the intervening unit productions

3. iteratively shorten long productions

   $$A \rightarrow B\ C\ \gamma \quad \Longrightarrow \quad \begin{cases} A \rightarrow X1\ \gamma \\ X1 \rightarrow B\ C \end{cases}$$

CF Grammars: Conversion to CNF

$\mathscr{L}_1$ **Grammar**	$\mathscr{L}_1$ **in CNF**
$S \rightarrow NP\ VP$	$S \rightarrow NP\ VP$
$S \rightarrow Aux\ NP\ VP$	$S \rightarrow X1\ VP$
	$X1 \rightarrow Aux\ NP$
$S \rightarrow VP$	$S \rightarrow book \mid include \mid prefer$
	$S \rightarrow Verb\ NP$
	$S \rightarrow X2\ PP$
	$S \rightarrow Verb\ PP$
	$S \rightarrow VP\ PP$
$NP \rightarrow Pronoun$	$NP \rightarrow I \mid she \mid me$
$NP \rightarrow Proper\text{-}Noun$	$NP \rightarrow TWA \mid Houston$
$NP \rightarrow Det\ Nominal$	$NP \rightarrow Det\ Nominal$
$Nominal \rightarrow Noun$	$Nominal \rightarrow book \mid flight \mid meal \mid money$
$Nominal \rightarrow Nominal\ Noun$	$Nominal \rightarrow Nominal\ Noun$
$Nominal \rightarrow Nominal\ PP$	$Nominal \rightarrow Nominal\ PP$
$VP \rightarrow Verb$	$VP \rightarrow book \mid include \mid prefer$
$VP \rightarrow Verb\ NP$	$VP \rightarrow Verb\ NP$
$VP \rightarrow Verb\ NP\ PP$	$VP \rightarrow X2\ PP$
	$X2 \rightarrow Verb\ NP$
$VP \rightarrow Verb\ PP$	$VP \rightarrow Verb\ PP$
$VP \rightarrow VP\ PP$	$VP \rightarrow VP\ PP$
$PP \rightarrow Preposition\ NP$	$PP \rightarrow Preposition\ NP$

Lexicon
$Det \rightarrow that \mid this \mid the \mid a$
$Noun \rightarrow book \mid flight \mid meal \mid money$
$Verb \rightarrow book \mid include \mid prefer$
$Pronoun \rightarrow I \mid she \mid me$
$Proper\text{-}Noun \rightarrow Houston \mid NWA$
$Aux \rightarrow does$
$Preposition \rightarrow from \mid to \mid on \mid near \mid through$

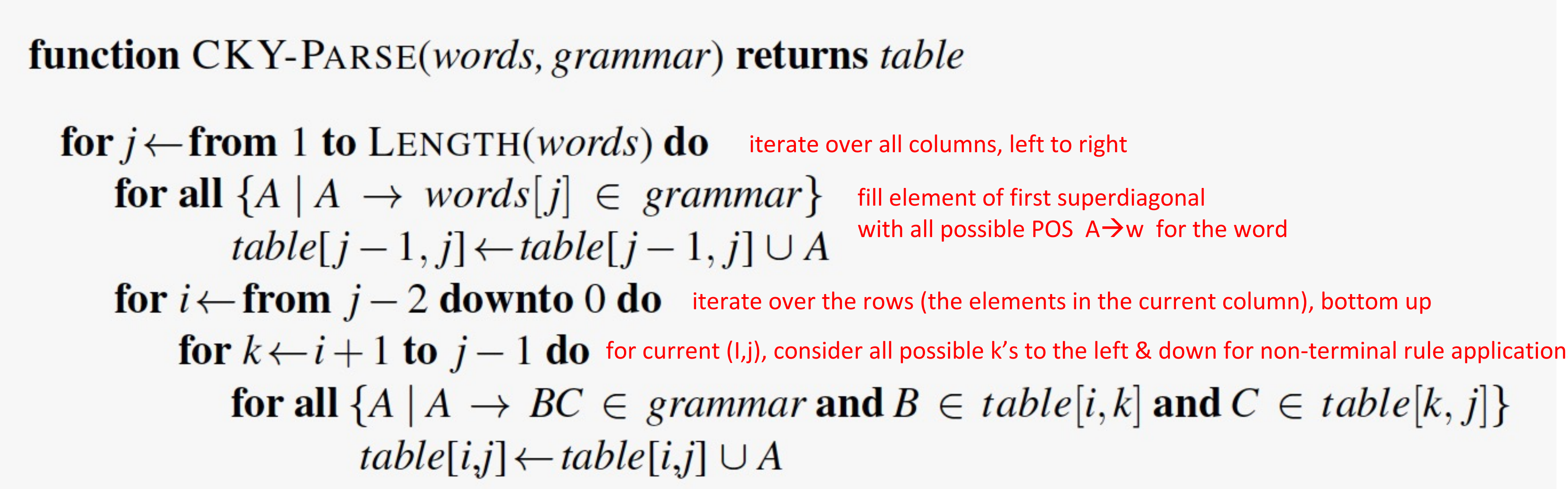
CYK Recognizer

- organize possible rule applications into upper triangle (i.e. excluding the diagonal) of an (n+1) x (n+1) matrix (n: sentence length)

- first super-diagonal (cells [0,1], [1,2], [2,3], …, [n-1,n]) contain all possible POS of the words of the sentence. (words are the terminals)

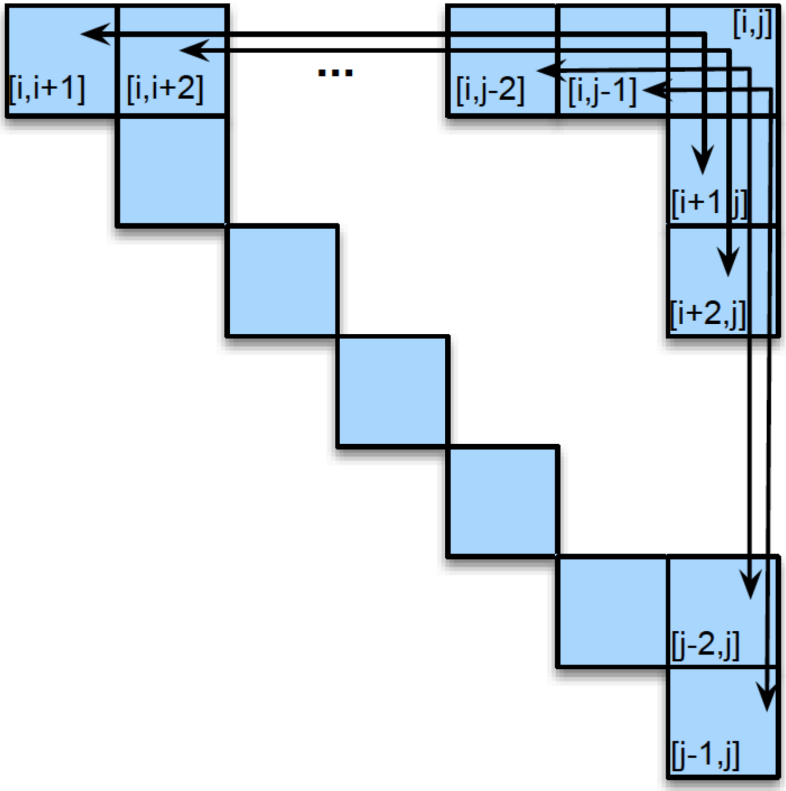
- other than first super-diagonal: cell [i,j] contains the set of all non-terminals that represent all possible constituents spanning positions i through j of the sentence (position-notation: *0 Book 1 that 2 flight 3 tomorrow 4*)

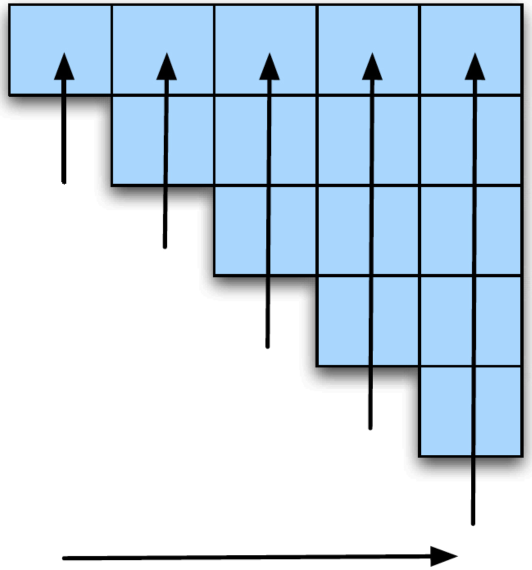
- **work** **up+right** from first super diagonal to left upper corner (cell [0,n]):
  fill cell [i,j] with all A from rules A → B C where
  B ∈ cell [i,k] (somwhere to the left) and C ∈ cell [k,j] (somewhere below) for all possible k (i<k<j).
  (for actual parser: also store k, B, C together with all possible multiple occurrences A_n of A in cell [i,j])



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

  for $j \leftarrow$ from 1 to LENGTH($words$) do          iterate over all columns, left to right
      for all $\{A \mid A \rightarrow words[j] \in grammar\}$      fill element of first superdiagonal
            $table[j-1,j] \leftarrow table[j-1,j] \cup A$        with all possible POS  A→w  for the word
      for $i \leftarrow$ from $j-2$ downto 0 do          iterate over the rows (the elements in the current column), bottom up
          for $k \leftarrow i+1$ to $j-1$ do          for current (I,j), consider all possible k's to the left & down for non-terminal rule applications
              for all $\{A \mid A \rightarrow BC \in grammar$ and $B \in table[i,k]$ and $C \in table[k,j]\}$
                  $table[i,j] \leftarrow table[i,j] \cup A$
```



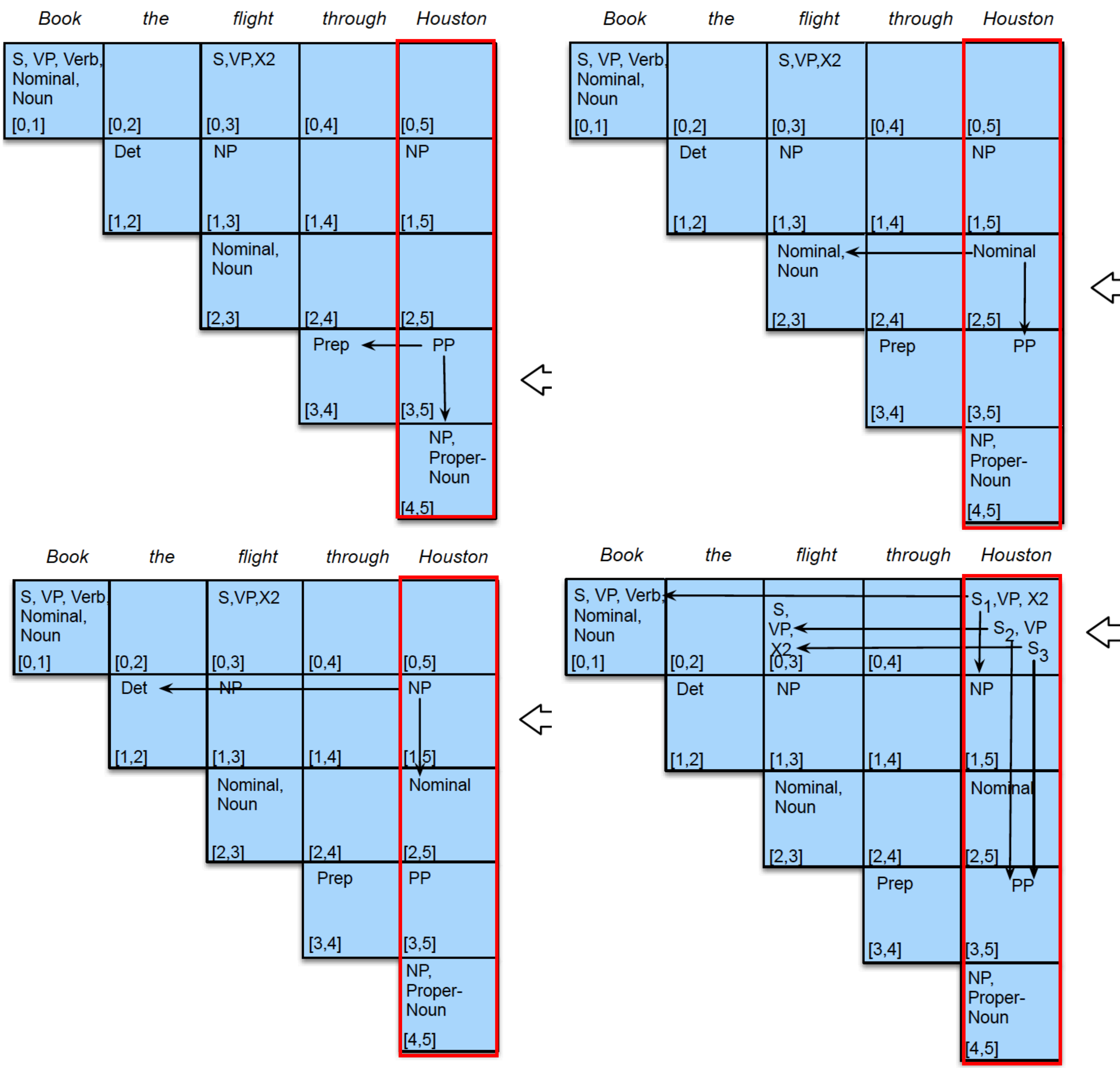


$\mathscr{L}_1$ in CNF

S → NP VP
S → X1 VP
X1 → Aux NP
S → book | include | prefer
S → Verb NP
S → X2 PP
S → Verb PP
S → VP PP
NP → I | she | me
NP → TWA | Houston
NP → Det Nominal
Nominal → book | flight | meal | money
Nominal → Nominal Noun
Nominal → Nominal PP
VP → book | include | prefer
VP → Verb NP
VP → X2 PP
X2 → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP

Lexicon

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



Book	the	flight	through	Houston
S, VP, Verb, Nominal, Noun [0,1]	[0,2]	S,VP,X2 [0,3]	[0,4]	[0,5]
	Det [1,2]	NP [1,3]	[1,4]	NP [1,5]
		Nominal, Noun [2,3]	[2,4]	[2,5]
			Prep [3,4]	PP [3,5]
				NP, Proper-Noun [4,5]

Book	the	flight	through	Houston
S, VP, Verb, Nominal, Noun [0,1]	[0,2]	S,VP,X2 [0,3]	[0,4]	[0,5]
	Det [1,2]	NP [1,3]	[1,4]	NP [1,5]
		Nominal, Noun [2,3]	[2,4]	Nominal [2,5]
			Prep [3,4]	PP [3,5]
				NP, Proper-Noun [4,5]

Book	the	flight	through	Houston
S, VP, Verb, Nominal, Noun [0,1]	[0,2]	S,VP,X2 [0,3]	[0,4]	[0,5]
	Det [1,2]	NP [1,3]	[1,4]	NP [1,5]
		Nominal, Noun [2,3]	[2,4]	Nominal [2,5]
			Prep [3,4]	PP [3,5]
				NP, Proper-Noun [4,5]

Book	the	flight	through	Houston
S, VP, Verb, Nominal, Noun [0,1]	[0,2]	S, VP, X2 [0,3]	[0,4]	S_1, VP, X2 S_2, VP S_3 [0,5]
	Det [1,2]	NP [1,3]	[1,4]	NP [1,5]
		Nominal, Noun [2,3]	[2,4]	Nominal [2,5]
			Prep [3,4]	PP [3,5]
				NP, Proper-Noun [4,5]

CYK Parser

- For every instance S_i of S in cell [0,n]: recursively repeat:
  - trace back all possible rules (productions) S → B C that led to this S_i
  - for each such rule, trace back all instances of B in its corresponding cell and trace back all instances of C in its corresponding cell
  - and so forth

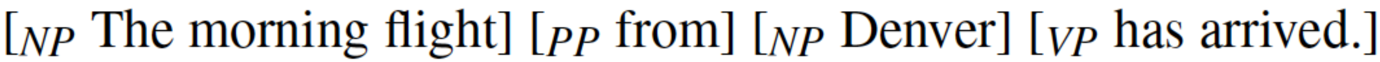
- For every instance S_i of S in cell [0,n] we may get an exponential number of possible parse trees

- resulting CNF parse trees may be linguistically unwieldy → convert back into original grammar or user adapted parsing algorithm that can parse non-CNF CF grammars

Chunking

- Chunking: identifying and classifying flat, non-overlapping segments of a sentence (chunks)

  chunks constitute the basic, non-recursive phrases corresponding to the major content bearing POS (noun phrases, verb phrases, adjective phrases, and prepositional phrases)



[$_{NP}$ The morning flight] [$_{PP}$ from] [$_{NP}$ Denver] [$_{VP}$ has arrived.]

- criteria for a chunk:
  - chunks do not contain other chunks of the same (or other) type
  - boundaries of a chunk: headword + pre-head modifiers;
    ignore post-head modifiers → assignment ambiguities due to overlap
    aka post-head(i-1) ∩ pre-head(i) are vastly reduced

Chunking as Classification Task

- Chunking: sequence classification (sequence labeling) task

- chunking: find limits for a chunk and classify it → BIO notation
  - for each of the n original labels: introduce
    - B version (beginning) and
    - I version (internal);
  - additional label: O (outside)
  - n original labels → 2n+1 new labels

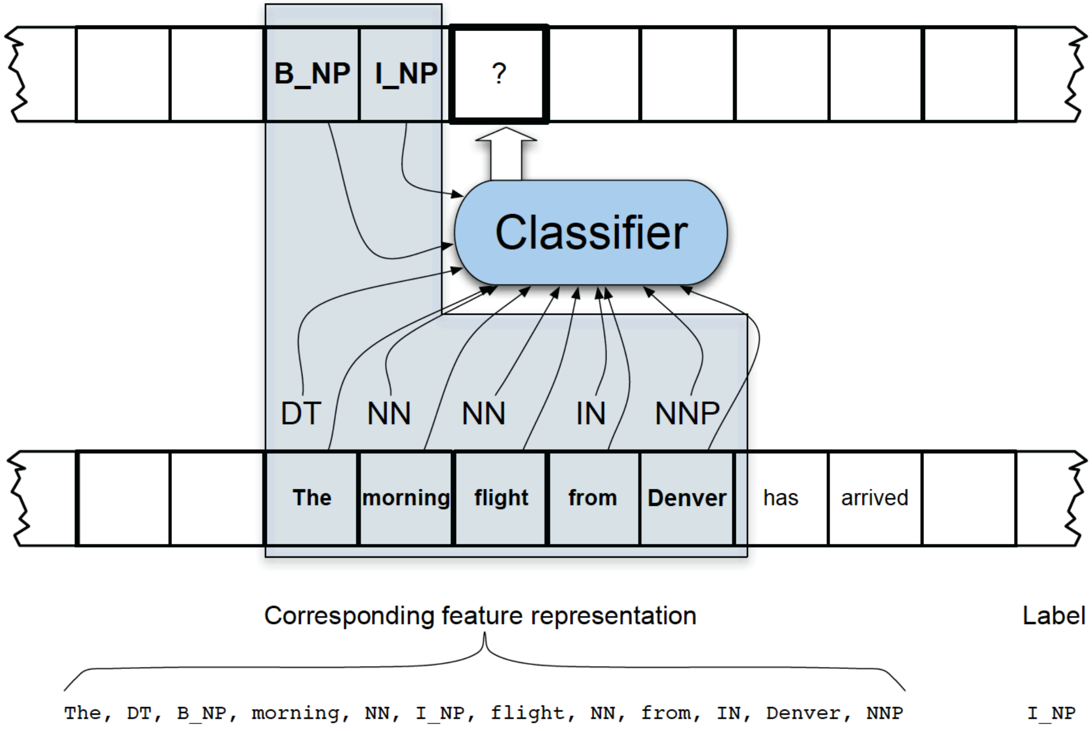
The	*morning*	*flight*	*from*	*Denver*	*has*	*arrived*
B_NP	I_NP	I_NP	B_PP	B_NP	B_VP	I_VP

The	*morning*	*flight*	*from*	*Denver*	*has*	*arrived.*
B_NP	I_NP	I_NP	O	B_NP	O	O

for an NP-only chunker

Chunking as Classification Task

- 2n+1 class labels;
  features per word in the sequence: from frame surrounding the word: e.g.
  - the word itself plus the two preceding words,
  - their parts-of-speech
  - the chunk tags of the preceding inputs in the window
  - ....


B_NP
I_NP
?
Classifier
DT
NN
NN
IN
NNP
The
morning
flight
from
Denver
has
arrived
Corresponding feature representation
Label
The, DT, B_NP, morning, NN, I_NP, flight, NN, from, IN, Denver, NNP
I_NP

Ground Truth for Chunking

- from parse trees in Treebanks:
  - concentrate on the non-terminals corresponding to the chunks that we are interested in (e.g. NP, VP)
  - boundary determination: in the respective phrase: find the head word (with head word detection rules) + all words preceding it; ignore words after head word

Bibliography

(1) Dan Jurafsky and James Martin: Speech and Language Processing (3rd ed. draft, version Oct 2019); Online: https://web.stanford.edu/~jurafsky/slp3/ (URL, Oct 2019); this slideset is especially based on chapter 13.

Recommendations for Studying

- minimal approach:

  work with the slides and understand their contents! Think beyond instead of merely memorizing the contents

- standard approach:

  minimal approach + read the corresponding pages in Jurafsky [1]

- interested students

  == standard approach