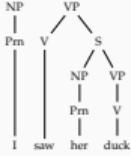


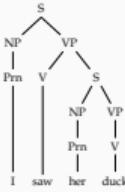


## Dealing with ambiguity



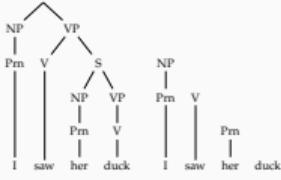
$S \rightarrow NP VP$   
 $NP \rightarrow Prn N$   
 $NP \rightarrow Prn$   
 $VP \rightarrow V NP$   
 $VP \rightarrow V$   
 $VP \rightarrow V S$   
 $N \rightarrow duck$   
 $V \rightarrow duck$   
 $V \rightarrow saw$   
 $Prn \rightarrow I$   
 $Prn \rightarrow she$   
 $Prn \rightarrow her$

## Dealing with ambiguity



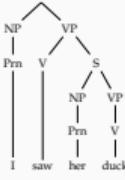
$S \rightarrow NP VP$   
 $NP \rightarrow Prn N$   
 $NP \rightarrow Prn$   
 $VP \rightarrow V NP$   
 $VP \rightarrow V$   
 $VP \rightarrow V S$   
 $N \rightarrow duck$   
 $V \rightarrow duck$   
 $V \rightarrow saw$   
 $Prn \rightarrow I$   
 $Prn \rightarrow she$   
 $Prn \rightarrow her$

## Dealing with ambiguity



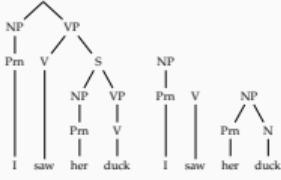
$S \rightarrow NP VP$   
 $NP \rightarrow Prn N$   
 $NP \rightarrow Prn$   
 $VP \rightarrow V NP$   
 $VP \rightarrow V$   
 $VP \rightarrow V S$   
 $N \rightarrow duck$   
 $V \rightarrow duck$   
 $V \rightarrow saw$   
 $Prn \rightarrow I$   
 $Prn \rightarrow she$   
 $Prn \rightarrow her$

## Dealing with ambiguity



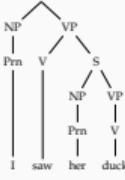
$S \rightarrow NP VP$   
 $NP \rightarrow Prn N$   
 $NP \rightarrow Prn$   
 $VP \rightarrow V NP$   
 $VP \rightarrow V$   
 $VP \rightarrow V S$   
 $N \rightarrow duck$   
 $V \rightarrow duck$   
 $V \rightarrow saw$   
 $Prn \rightarrow I$   
 $Prn \rightarrow she$   
 $Prn \rightarrow her$

## Dealing with ambiguity



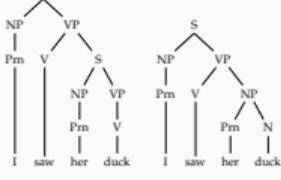
$S \rightarrow NP VP$   
 $NP \rightarrow Prn N$   
 $NP \rightarrow Prn$   
 $VP \rightarrow V NP$   
 $VP \rightarrow V$   
 $VP \rightarrow V S$   
 $N \rightarrow duck$   
 $V \rightarrow duck$   
 $V \rightarrow saw$   
 $Prn \rightarrow I$   
 $Prn \rightarrow she$   
 $Prn \rightarrow her$

## Dealing with ambiguity



$S \rightarrow NP VP$   
 $NP \rightarrow Prn N$   
 $NP \rightarrow Prn$   
 $VP \rightarrow V NP$   
 $VP \rightarrow V$   
 $VP \rightarrow V S$   
 $N \rightarrow duck$   
 $V \rightarrow duck$   
 $V \rightarrow saw$   
 $Prn \rightarrow I$   
 $Prn \rightarrow she$   
 $Prn \rightarrow her$

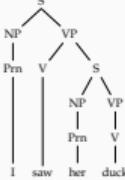
## Dealing with ambiguity



$S \rightarrow NP VP$   
 $NP \rightarrow Prn N$   
 $NP \rightarrow Prn$   
 $VP \rightarrow V NP$   
 $VP \rightarrow V$   
 $VP \rightarrow V S$   
 $N \rightarrow duck$   
 $V \rightarrow duck$   
 $V \rightarrow saw$   
 $Prn \rightarrow I$   
 $Prn \rightarrow she$   
 $Prn \rightarrow her$

## How to represent multiple parses

parse forest grammar



$S_{0,4} \rightarrow NP_{0,1} VP_{1,4}$   
 $NP_{0,1} \rightarrow Prn_{0,1}$   
 $Prn_{0,1} \rightarrow I_{0,1}$   
 $I_{0,1} \rightarrow saw_{1,2}$   
 $saw_{1,2} \rightarrow her_{2,3}$   
 $her_{2,3} \rightarrow duck_{3,4}$   
 $V_{1,4} \rightarrow V_{1,2} S_{2,4}$   
 $S_{2,4} \rightarrow Prn_{2,3} V_{3,4}$   
 $V_{3,4} \rightarrow duck_{3,4}$   
 $V_{1,4} \rightarrow V_{1,2} NP_{2,4}$   
 $NP_{2,4} \rightarrow Prn_{2,3} N_{3,4}$   
 $N_{3,4} \rightarrow duck_{3,4}$

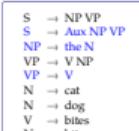
## CKY algorithm

- The CKY (Cocke-Kasami-Younger) parsing algorithm is a dynamic programming algorithm
- It processes the input bottom up, and saves the intermediate results on a chart
- Time complexity for recognition is  $O(n^3)$
- Space complexity is  $O(n^2)$
- It requires the CFG to be in *Chomsky normal form* (CNF) (can somewhat be relaxed, but not common)

## Chomsky normal form (CNF)

- A CFG is in CNF, if the rewrite rules are in one of the following forms
  - $A \rightarrow B C$
  - $A \rightarrow a$
 where  $A, B, C$  are non-terminals and  $a$  is a terminal
- Any CFG can be converted to CNF
- Resulting grammar is usually equivalent to the original grammar:
  - it generates/accepts the same language
  - but the derivations are different

## Converting to CNF: example



$S \rightarrow NP VP$   
 $S \rightarrow Aux NP VP$   
 $NP \rightarrow the N$   
 $NP \rightarrow V NP$   
 $VP \rightarrow V$   
 $N \rightarrow cat$   
 $N \rightarrow dog$   
 $V \rightarrow bites$   
 $N \rightarrow bites$

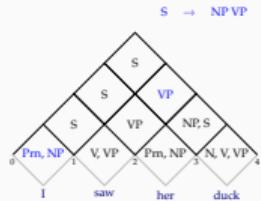
## Converting to CNF

- Eliminate the  $c$  rules: if  $A \rightarrow c$  is in the grammar
  - replace any rule  $B \rightarrow a A \beta$  with two rules  
 $B \rightarrow a \beta$   
 $B \rightarrow a \beta$
  - add  $A' \rightarrow a$  for all  $a$  (except  $c$ ) whose LHS is  $A$
  - repeat the process for newly created  $c$  rules
    - remove the rules with  $c$  on the RHS (except  $S \rightarrow c$ )
- Eliminate unit rules: for a rule  $A \rightarrow B$ 
  - Replace the rule with  $A \rightarrow a_1 | \dots | a_n$ , where  $a_1, \dots, a_n$  are all RHS or rule B
  - Remove the rule  $A \rightarrow B$
  - Repeat the process until no unit rules remain
- Binaryize all the non-binary rules with non-terminal on the RHS: for a rule  $A \rightarrow X_1 X_2 \dots X_n$ 
  - Replace the rule with  $A \rightarrow A_1 X_1 \dots X_n$ , and  $A_1 \rightarrow X_1 X_2$
  - Repeat the process until all new rules are binary



## CKY demonstration

an ambiguous example



© Cölkner, SIR / University of Tübingen

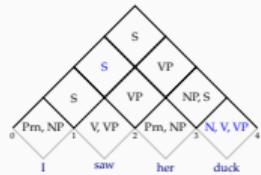
Winter Semester 2020/21 9 / 13

$S \rightarrow NP VP$   
 $NP \rightarrow Prn N$   
 $VP \rightarrow V NP$   
 $V \rightarrow V S$   
 $N \rightarrow duck$   
 $VP \rightarrow duck | saw$   
 $V \rightarrow duck | saw$   
 $Prn \rightarrow I | she | her$   
 $NP \rightarrow I | she | her$

Winter Semester 2020/21 9 / 13

## CKY demonstration

an ambiguous example



© Cölkner, SIR / University of Tübingen

Winter Semester 2020/21 10 / 13

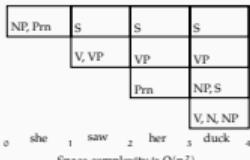
Winter Semester 2020/21 10 / 13

$S \rightarrow NP VP$   
 $NP \rightarrow Prn N$   
 $VP \rightarrow V NP$   
 $V \rightarrow V S$   
 $N \rightarrow duck$   
 $VP \rightarrow duck | saw$   
 $V \rightarrow duck | saw$   
 $Prn \rightarrow I | she | her$   
 $NP \rightarrow I | she | her$

© Cölkner, SIR / University of Tübingen

## CKY demonstration: the chart

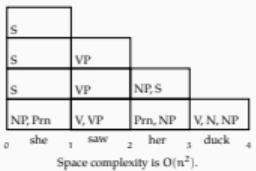
our chart is a 2D array

Space complexity is  $O(n^2)$ .

Winter Semester 2020/21 10 / 13

## CKY demonstration: the chart

our chart is a 2D array – this is more convenient for programming

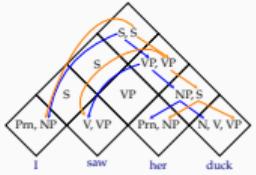
Space complexity is  $O(n^2)$ .

© Cölkner, SIR / University of Tübingen

Winter Semester 2020/21 11 / 13

Winter Semester 2020/21 11 / 13

## Chart parsing example (CKY parsing)



The chart stores a parse forest efficiently.

© Cölkner, SIR / University of Tübingen

Winter Semester 2020/21 12 / 13

Winter Semester 2020/21 12 / 13

## Parsing vs. recognition

- We went through a recognition example
- Note that the algorithm is not directional: it takes the complete input
- Recognition accepts or rejects a sentence based on a grammar
- For parsing, we want to know the derivations that yielded a correct parse
- To recover parse trees, we
  - follow the same procedure as recognition
  - add back links to keep track of the derivations

## Summary

- CKY avoids re-computing the analyses by storing the earlier analyses (of sub-spans) in a table
- It still computes lower level constituents that are not allowed by the grammar
- CKY requires the grammar to be in CNF
- CKY has  $O(n^4)$  recognition complexity
- For parsing we need to keep track of backlinks
- CKY can efficiently store all possible parses in a chart
- Enumerating all possible parses have exponential complexity (worst case)
- Suggested reading: Jurafsky and Martin (chapter 18 2009, 3rd ed draft)

Next:

- Top-down chart parsing: Earley algorithm
- Suggested reading:
  - Grune and Jacobs (2007, section 7.2)

© Cölkner, SIR / University of Tübingen

Winter Semester 2020/21 13 / 13

## Acknowledgments, references, additional reading material

Grune, Dick and Csepe, D. (2007). *Parsing Techniques: A Practical Guide* (second, Monographs in Computer Science). The first edition is available at [http://www.cs.rug.nl/~bernie/PTAGS\\_1st\\_edition/berniebook.pdf](http://www.cs.rug.nl/~bernie/PTAGS_1st_edition/berniebook.pdf). Springer New York, ISBN 9780387340518.

Joshi, Aravind and Jones, H. Martin (2010). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (second edition). Pearson Pearson Hall, ISBN 9780131033905.

© Cölkner, SIR / University of Tübingen

Winter Semester 2020/21 A.1