

Computerlinguistik II / Sprachtechnologie

Vorlesung im SS 2010
(M-GSW-10)

Prof. Dr. Udo Hahn

Lehrstuhl für Computerlinguistik
Institut für Germanistische Sprachwissenschaft
Friedrich-Schiller-Universität Jena

der folgende Teil der Vorlesung
ist einer Ausarbeitung entnommen von

Dr. Christel Kemke

Assistant Professor
Department of Computer Science
University of Manitoba

Sample Grammar

Grammar (S, NT, T, P) – Sentence Symbol $S \in NT$, Part-of-Speech $\subseteq NT$,
syntactic Constituents $\subseteq NT$, Grammar Rules $P \subseteq NT \times (NT \cup T)^*$

$S \rightarrow NP VP$ statement

$S \rightarrow Aux NP VP$ question

$S \rightarrow VP$ command

$NP \rightarrow Det Nominal$

$NP \rightarrow Proper-Noun$

$Nominal \rightarrow Noun \mid Noun Nominal \mid Nominal PP$

$VP \rightarrow Verb \mid Verb NP \mid Verb PP \mid Verb NP PP$

$PP \rightarrow Prep NP$

$Det \rightarrow that \mid this \mid a$

$Noun \rightarrow book \mid flight \mid meal \mid money$

$Proper-Noun \rightarrow Houston \mid American Airlines \mid TWA$

$Verb \rightarrow book \mid include \mid prefer$

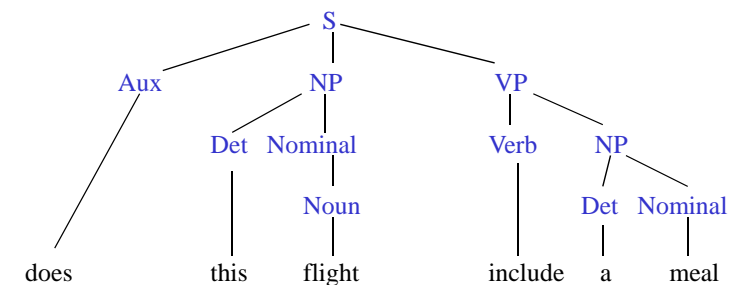
$Aux \rightarrow does$

$Prep \rightarrow from \mid to \mid on$

Task: Parse "Does this flight include a meal?"

Sample Parse Tree

Task: Parse "Does this flight include a meal?"



Problems in Parsing - Ambiguity

Ambiguity

“One morning, I shot an elephant in my pajamas.

How he got into my pajamas, I don’t know.”

Groucho Marx

syntactical/structural ambiguity – several parse trees are possible e.g. above sentence

semantic/lexical ambiguity – several word meanings e.g. bank (where you get money) and (river) bank

even **different word categories** possible (interim) e.g. “He **books** the flight.” vs. “The **books** are here.” or “Fruit flies from the balcony” vs. “Fruit flies are on the balcony.”

Problems in Parsing - Attachment

Attachment

in particular **PP (prepositional phrase) binding**; often referred to as ‘**binding problem**’

“One morning, I shot an elephant in my pajamas.”

(S ... (NP (PNoun I)(VP (Verb shot) (NP (Det an (Nominal (Noun elephant)))) (PP in my pajamas))...)

rule **VP → Verb NP PP**

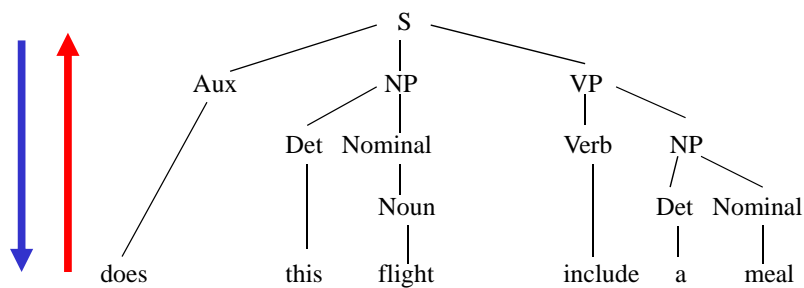
(S ... (NP (PNoun I)) (VP (Verb shot) (NP (Det an) (Nominal (Nominal (Noun elephant) (PP in my pajamas)...)

rule **VP → Verb NP** and **NP → Det Nominal** and **Nominal → Nominal PP** and **Nominal → Noun**

Bottom-up and Top-down Parsing

Bottom-up – from word-nodes to sentence-symbol

Top-down Parsing – from sentence-symbol to words



Problems with Bottom-up and Top-down Parsing

Problems with **left-recursive rules** like **NP → NP PP**: don’t know how many times recursion is needed

Pure Bottom-up or Top-down Parsing is **inefficient** because it generates and explores too many structures which in the end turn out to be invalid (several grammar rules applicable → ‘interim’ ambiguity).

Combine top-down and bottom-up approach:

Start with sentence; use rules top-down (**look-ahead**); read input; try to find shortest path from input to highest unparsed constituent (from **left to right**).

→ **Chart-Parsing / Earley-Parser**

Chart Parsing / Earley Algorithm

Earley-Parser based on Chart-Parsing

Essence: Integrate top-down and bottom-up parsing. Keep recognized sub-structures (sub-trees) for shared use during parsing.

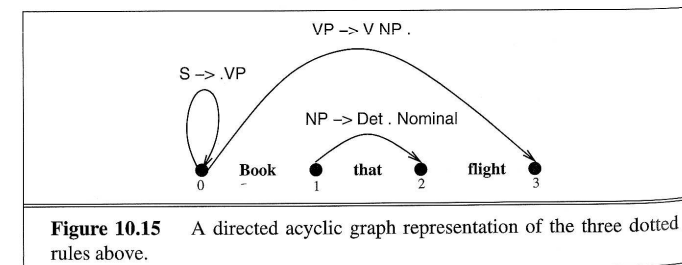
Top-down: Start with S-symbol. Generate all applicable rules for S. Go further down with left-most constituent in rules and add rules for these constituents until you encounter a left-most node on the RHS which is a word category (POS).

Bottom-up: Read input word and compare. If word matches, mark as recognized and move parsing on to the next category in the rule(s).

Chart

Chart

Sequence of n input words; $n+1$ nodes marked 0 to n . Arcs indicate recognized part of RHS of rule. The \bullet indicates recognized constituents in rules.



Jurafsky & Martin, Figure 10.15, p. 380

Chart Parsing / Earley Parser 1

Chart

Sequence of input words; $n+1$ nodes marked 0 to n . States in chart represent possible rules and recognized constituents, with arcs.

Interim state

$S \rightarrow \bullet VP, [0,0]$

- top-down look at rule $S \rightarrow VP$
- nothing of RHS of rule yet recognized (\bullet is far left)
- arc at beginning, no coverage (covers no input word; beginning of arc at 0 and end of arc at 0)

Chart Parsing / Earley Parser 2

Interim states

$NP \rightarrow Det \bullet Nominal, [1,2]$

- top-down look with rule $NP \rightarrow Det \bullet Nominal$
- Det recognized (\bullet after Det)
- arc covers one input word which is between node 1 and node 2
- look next for Nominal

$NP \rightarrow Det Nominal \bullet, [1,3]$

- Nominal was recognized, move \bullet after Nominal
- move end of arc to cover Nominal (change 2 to 3)
- structure is completely recognized; arc is inactive; mark NP as recognized in other rules (move \bullet).

Chart - 0

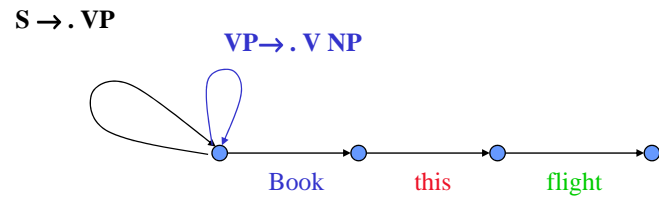


Chart - 1

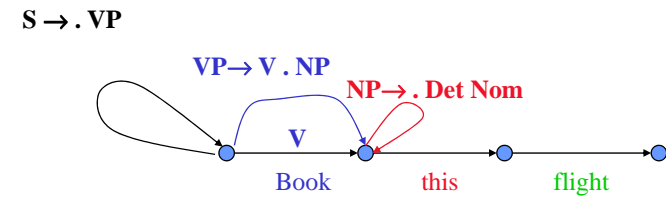


Chart - 2

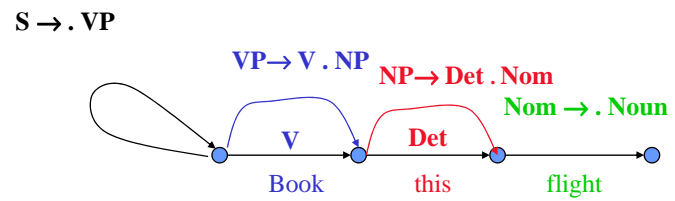


Chart - 3a

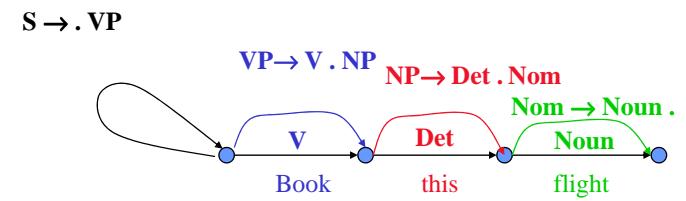


Chart - 3b

$S \rightarrow \cdot VP$

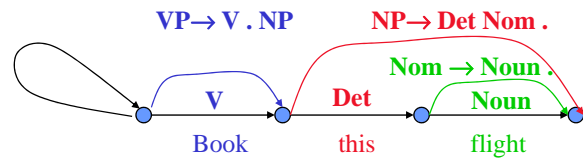


Chart - 3c

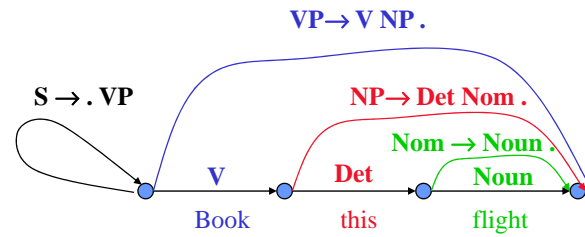


Chart - 3d

$S \rightarrow VP \cdot$

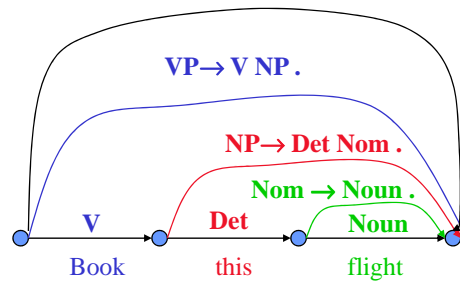


Chart - All States

$S \rightarrow VP \cdot$

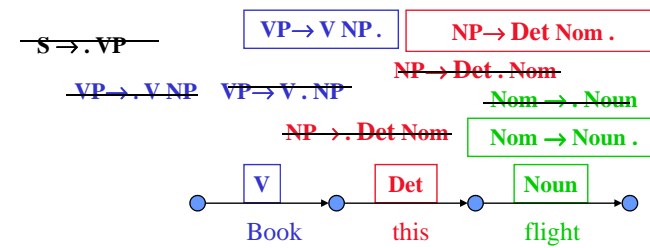


Chart - Final States

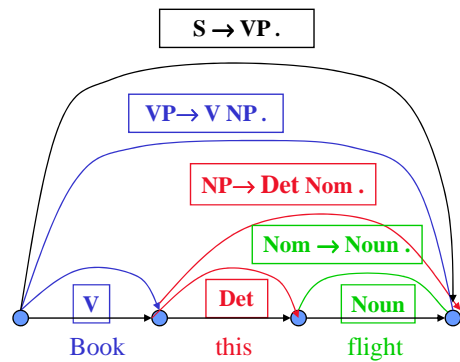


Chart 0 with two S- and two VP-Rules

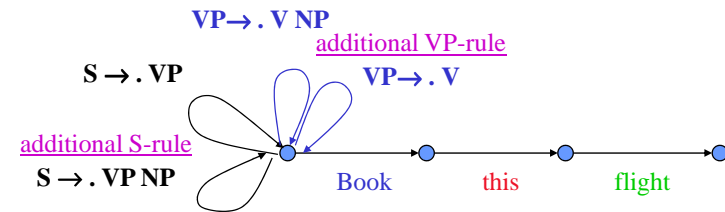


Chart 1a with two S- and two VP-Rules

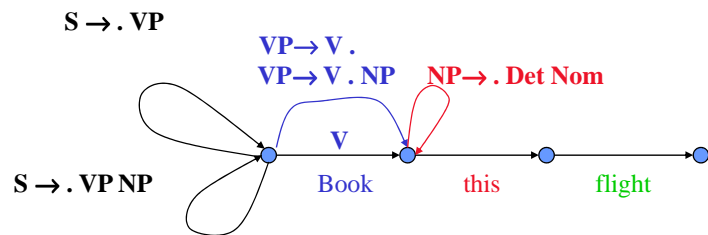


Chart 1b with two S- and two VP-Rules

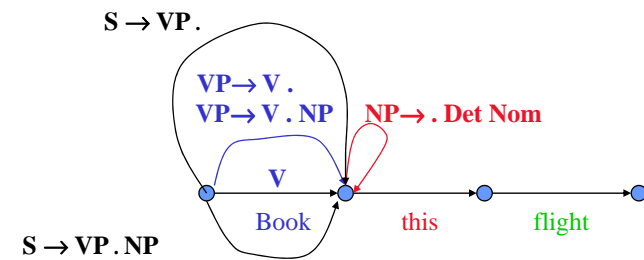


Chart 2 with two S- and two VP-Rules

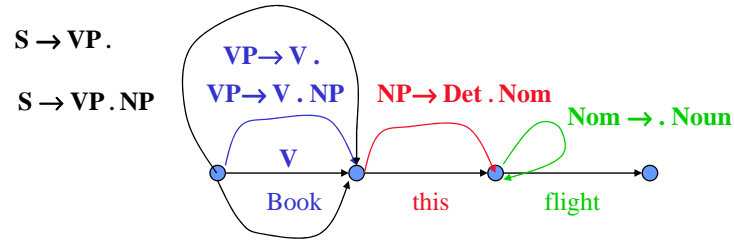
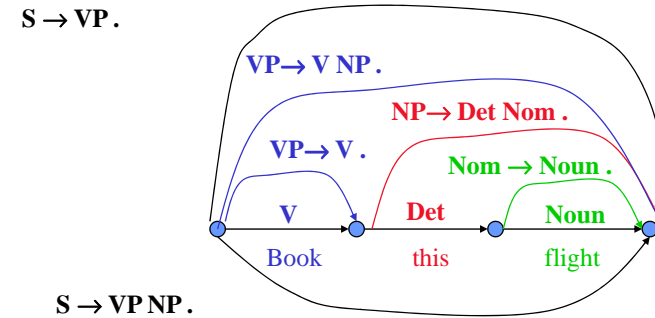
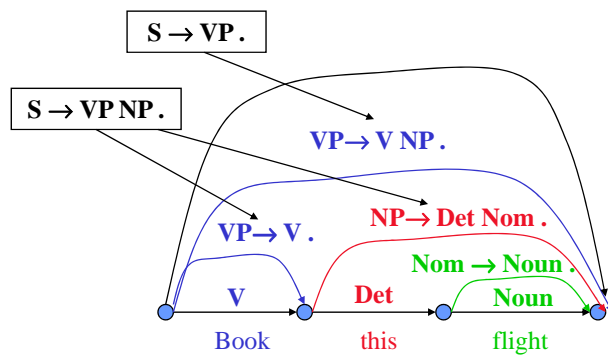


Chart 3 with two S- and two VP-Rules



Final Chart - with two S-and two VP-Rules



Earley Algorithm - Functions

predictor

generates new rules for partly recognized RHS with constituent right of • (**top-down generation**)

scanner

if word category (POS) is found right of the •, the Scanner reads the next input word and adds a rule for it to the chart (**bottom-up mode**)

completer

if rule is completely recognized (the • is far right), the recognition state of earlier rules in the chart advances: the • is moved over the recognized constituent (**bottom-up recognition**).

Earley-Algorithm

```
function EARLEY-PARSE(words, grammar) returns chart
  ENQUEUE( $(\gamma \rightarrow \bullet S, [0,0])$ , chart[0])
  for  $i$  from 0 to LENGTH(words) do
    for each state in chart[i] do
      if INCOMPLETE?(state) and
        NEXT-CAT(state) is not a part of speech
      then PREDICTOR(state)
      elseif INCOMPLETE?(state) and
        NEXT-CAT(state) is a part of speech
      then SCANNER(state)
      else COMPLETER(state)
    end
  end
  return(chart)
```

- continued on next slide -

```
procedure PREDICTOR( $(A \rightarrow \alpha \bullet B \beta, [i, j])$ )
  for each  $(B \rightarrow \gamma)$  in GRAMMAR-RULES-FOR( $B, grammar$ )
  do ENQUEUE( $(B \rightarrow \gamma [j, j], chart[j])$ )
end

procedure SCANNER ( $(A \rightarrow \alpha \bullet B \beta, [i, j])$ )
  if  $B \in$  PARTS-OF-SPEECH(word[j])
  then ENQUEUE( $(B \rightarrow word[j], [j, j+1], chart[j+1])$ )

procedure COMPLETER ( $(B \rightarrow \gamma \bullet, [j, k])$ )
  for each  $(A \rightarrow \alpha \bullet B \beta, [i, j])$  in chart[j]
  do ENQUEUE( $(A \rightarrow \alpha B \bullet \beta, [i, k], chart[k])$ )
end

procedure ENQUEUE(state, chart-entry)
  if state is not already in chart-entry
  then PUSH(state, chart-entry)
end
```

Earley Algorithm

- Figures -

Jurafsky & Martin, Figures 10.16, 10.17, 10.18

```
function EARLEY-PARSE(words, grammar) returns chart
  ENQUEUE( $(\gamma \rightarrow \bullet S, [0,0], chart[0])$ )
  for  $i$  from 0 to LENGTH(words) do
    for each state in chart[i] do
      if INCOMPLETE?(state) and
        NEXT-CAT(state) is not a part of speech then
        PREDICTOR(state)
      elseif INCOMPLETE?(state) and
        NEXT-CAT(state) is a part of speech then
        SCANNER(state)
      else
        COMPLETER(state)
      end
    end
  end
  return(chart)

procedure PREDICTOR( $(A \rightarrow \alpha \bullet B \beta, [i, j])$ )
  for each  $(B \rightarrow \gamma)$  in GRAMMAR-RULES-FOR( $B, grammar$ ) do
    ENQUEUE( $(B \rightarrow \gamma, [j, j], chart[j])$ )
  end

procedure SCANNER( $(A \rightarrow \alpha \bullet B \beta, [i, j])$ )
  if  $B \in$  PARTS-OF-SPEECH(word[j]) then
    ENQUEUE( $(B \rightarrow word[j], [j, j+1], chart[j+1])$ )

procedure COMPLETER( $(B \rightarrow \gamma \bullet, [j, k])$ )
  for each  $(A \rightarrow \alpha \bullet B \beta, [i, j])$  in chart[j] do
    ENQUEUE( $(A \rightarrow \alpha B \bullet \beta, [i, k], chart[k])$ )
  end

procedure ENQUEUE(state, chart-entry)
  if state is not already in chart-entry then
    PUSH(state, chart-entry)
  end
```

Chart[0]		
$\gamma \rightarrow \bullet S$	[0,0]	Dummy start state
$S \rightarrow \bullet NP VP$	[0,0]	Predictor
$NP \rightarrow \bullet Det NOMINAL$	[0,0]	Predictor
$NP \rightarrow \bullet Proper-Noun$	[0,0]	Predictor
$S \rightarrow \bullet Aux NP VP$	[0,0]	Predictor
$S \rightarrow \bullet VP$	[0,0]	Predictor
$VP \rightarrow \bullet Verb$	[0,0]	Predictor
$VP \rightarrow \bullet Verb NP$	[0,0]	Predictor

Chart[1]		
$Verb \rightarrow book \bullet$	[0,1]	Scanner
$VP \rightarrow Verb \bullet$	[0,1]	Completer
$S \rightarrow VP \bullet$	[0,1]	Completer
$VP \rightarrow Verb \bullet NP$	[0,1]	Completer
$NP \rightarrow \bullet Det NOMINAL$	[1,1]	Predictor
$NP \rightarrow \bullet Proper-Noun$	[1,1]	Predictor

Chart[2]		
$Det \rightarrow that \bullet$	[1,2]	Scanner
$NP \rightarrow Det \bullet NOMINAL$	[1,2]	Completer
$NOMINAL \rightarrow \bullet Noun$	[2,2]	Predictor
$NOMINAL \rightarrow \bullet Noun NOMINAL$	[2,2]	Predictor

Chart[3]		
$Noun \rightarrow flight \bullet$	[2,3]	Scanner
$NOMINAL \rightarrow Noun \bullet$	[2,3]	Completer
$NOMINAL \rightarrow Noun \bullet NOMINAL$	[2,3]	Completer
$NP \rightarrow Det NOMINAL \bullet$	[1,3]	Completer
$VP \rightarrow Verb NP \bullet$	[0,3]	Completer
$S \rightarrow VP \bullet$	[0,3]	Completer
$NOMINAL \rightarrow \bullet Noun$	[3,3]	Predictor
$NOMINAL \rightarrow \bullet Noun NOMINAL$	[3,3]	Predictor

Chart[0]		
S0 $\gamma \rightarrow \bullet S$	[0,0]	[] Dummy start state
S1 $S \rightarrow \bullet NP VP$	[0,0]	[] Predictor
S2 $NP \rightarrow \bullet Det NOMINAL$	[0,0]	[] Predictor
S3 $NP \rightarrow \bullet Proper-Noun$	[0,0]	[] Predictor
S4 $S \rightarrow \bullet Aux NP VP$	[0,0]	[] Predictor
S5 $S \rightarrow \bullet VP$	[0,0]	[] Predictor
S6 $VP \rightarrow \bullet Verb$	[0,0]	[] Predictor
S7 $VP \rightarrow \bullet Verb NP$	[0,0]	[] Predictor

Chart[1]		
S8 $Verb \rightarrow book \bullet$	[0,1]	[] Scanner
S9 $VP \rightarrow Verb \bullet$	[0,1]	[S8] Completer
S10 $S \rightarrow VP \bullet$	[0,1]	[S9] Completer
S11 $VP \rightarrow Verb \bullet NP$	[0,1]	[S8] Completer
S12 $NP \rightarrow \bullet Det NOMINAL$	[1,1]	[] Predictor
S13 $NP \rightarrow \bullet Proper-Noun$	[1,1]	[] Predictor

Chart[2]		
S14 $Det \rightarrow that \bullet$	[1,2]	[] Scanner
S15 $NP \rightarrow Det \bullet NOMINAL$	[1,2]	[S14] Completer
S16 $NOMINAL \rightarrow \bullet Noun$	[2,2]	[] Predictor
S17 $NOMINAL \rightarrow \bullet Noun NOMINAL$	[2,2]	[] Predictor

Chart[3]		
S18 $Noun \rightarrow flight \bullet$	[2,3]	[] Scanner
S19 $NOMINAL \rightarrow Noun \bullet$	[2,3]	[S18] Completer
S20 $NOMINAL \rightarrow Noun \bullet NOMINAL$	[2,3]	[S18] Completer
S21 $NP \rightarrow Det NOMINAL \bullet$	[1,3]	[S14,S19] Completer
S22 $VP \rightarrow Verb NP \bullet$	[0,3]	[S8,S21] Completer
S23 $S \rightarrow VP \bullet$	[0,3]	[S22] Completer
S24 $NOMINAL \rightarrow \bullet Noun$	[3,3]	[] Predictor
S25 $NOMINAL \rightarrow \bullet Noun NOMINAL$	[3,3]	[] Predictor

Additional References

Jurafsky, D. & J. H. Martin, *Speech and Language Processing*, Prentice-Hall, 2000. (Chapters 9 and 10)

Earley Algorithm

Jurafsky & Martin, Figure 10.16, p.384

Earley Algorithm - Examples

Jurafsky & Martin, Figures 10.17 and 10.18