Introduction to NLP

Syntax and Parsing. Context Free Grammars

Alexandre Rademaker [1]

FGV/EMAp

September 27, 2022

[1] Olga Zamaraeva

Overview

- Parsing: constituent and dependency
- Leonard Bernstein's musical syntax
  - `https://www.youtube.com/watch?v=r_fxB6yrDVo`
- Target representations
- Context Free Grammars
- Evaluating parsing
- Treebanking

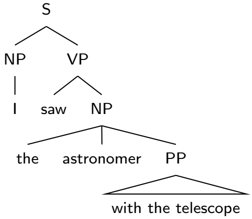
Parsing

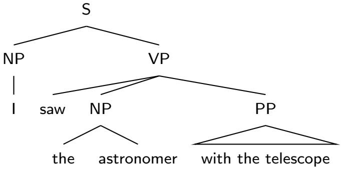
- Recognizing string as input and assigning structure to it
- Syntactic parsing: assigning syntactic structure
- Semantic parsing: assigning semantic structure

Syntactic Parsing

Parsing: Making explicit structure that is inherent (implicit) in natural language strings

- What is that structure?
- Why would we need it?

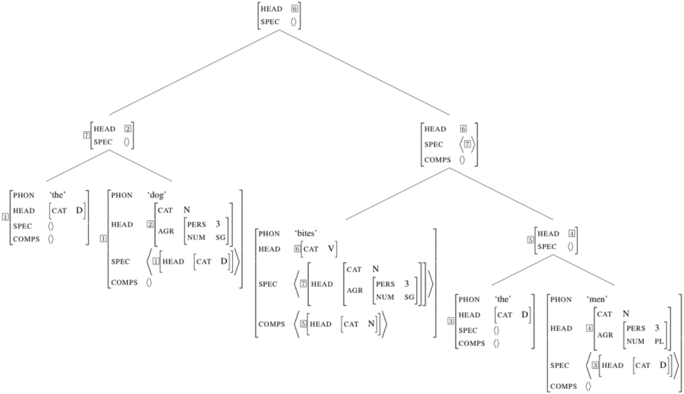




Parsing

Parsing: Making explicit structure that is inherent (implicit) in natural language strings

- What is that structure?
- Why would we need it?



pic from: `http://www.dobnik.net/simon/teaching/shared/LT2112-formling/pics/?MA`

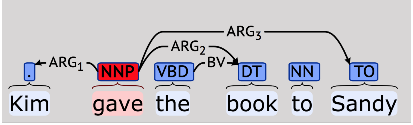
Implicit structure

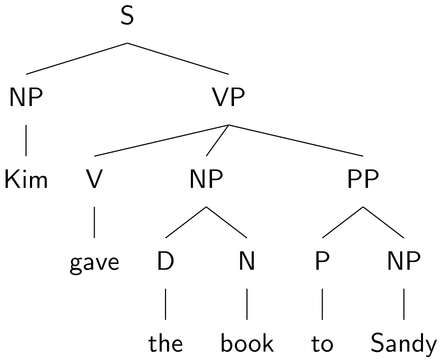
- What do these sentences have in common?
  - Kim gave the book to Sandy.
  - Kim gave Sandy the book.
  - The book was given to Sandy by Kim.
  - This is the book that Kim gave to Sandy.
  - Which book did Kim give to Sandy?
  - Kim will be expected to continue to try to give the book to Sandy.
  - This book everyone agrees Pat thinks Kim gave to Sandy.
  - This book is difficult for Kim to give to Sandy.

Implicit structure: Constituent structure & Dependency structure

Kim gave the book to Sandy.

- (S (NP Kim) (VP (V gave) (NP (D the) (N book)) (PP (P to) (NP Sandy))))
- subj(gave, Kim); dobj(gave, book); iobj(gave, to); dobj(to, Sandy); spec(book, the)



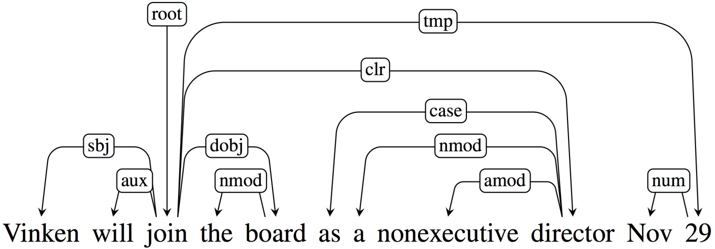


Dependency parsing

- Instead of constituents, look at grammatical relations between heads of constituents
  - Why?
- flexible word order
- semantics
- relations between active and passive
- Harder to explain the **ungrammaticality** of certain constructions
  - lack of theoretical ground

Dependency structure

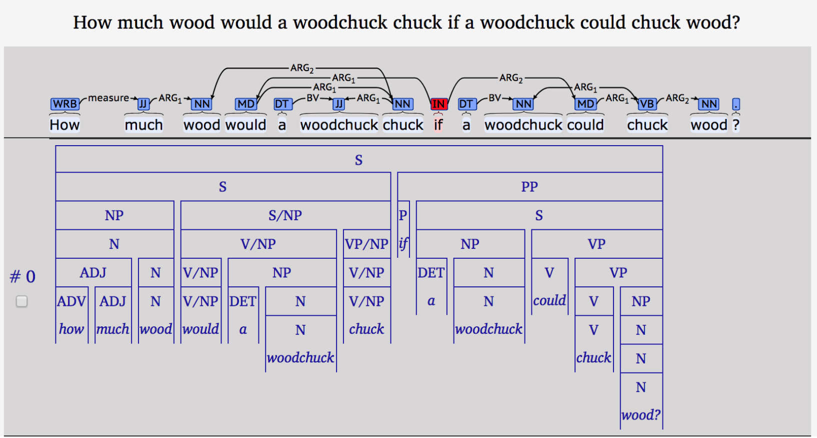
(14.4)


root
tmp
clr
case
sbj
dobj
nmod
aux
nmod
amod
num
Vinken will join the board as a nonexecutive director Nov 29

Exercise: Constituent Structure & Dependency Structure

How much wood would a woodchuck chuck if a woodchuck could chuck wood?
See `https://youtu.be/pF0RgB1dZTU`.

English Resource Grammar



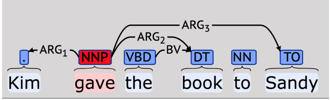
How much wood would a woodchuck chuck if a woodchuck could chuck wood?

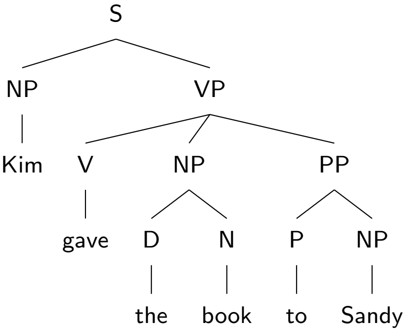
Stanford Parser

```
(ROOT
  (SBARQ
    (WHNP
      (WHADJP (WRB How) (JJ much))
      (NNS wood))
    (SQ (MD would)
      (NP (DT a) (NN woodchuck))
      (VP (VB chuck)
        (SBAR (IN if)
          (S
            (NP (DT a) (NN woodchuck))
            (VP (MD could)
              (VP (VB chuck)
                (NP (NN wood))))))))
```

When do we need structure?

- When do we need constituent structure?
- When do we need dependency structure?





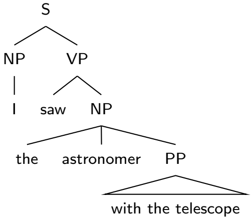
When do we need structure?

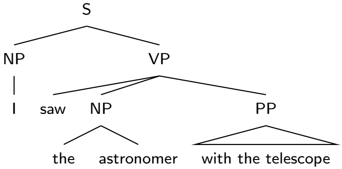
- When do we need constituent structure?
  - Structured language models (ASR, MT)
  - Translation models (MT)
  - Generation
  - TTS: assigning intonation information
- When do we need dependency structure?
  - Information extraction (... QA, machine reading)
  - Dialogue systems
  - Sentiment analysis
  - Transfer-based MT

Ambiguity

Parsing: Making explicit the structure that is inherent (implicit) in natural language strings

- How does it relate to the ambiguity issue?
- Suppose you have a parser. Does it help you with ambiguity?

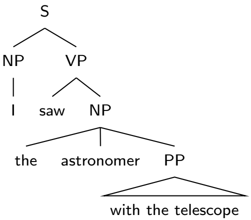


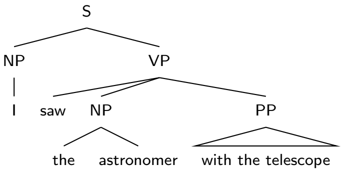


Ambiguity

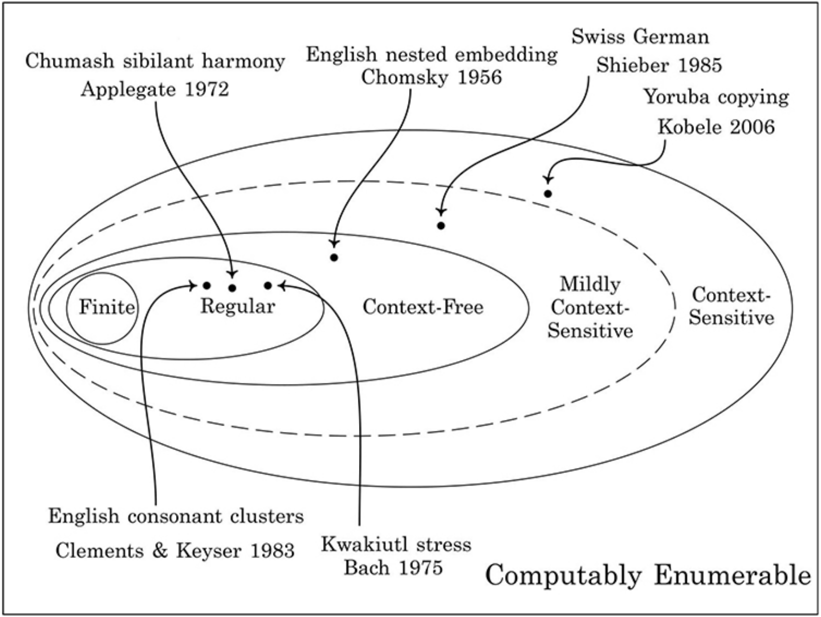
Parsing: Making explicit structure that is inherent (implicit) in natural language strings

- How does it relate to the ambiguity issue?
- Suppose you have a parser. Does it help you with ambiguity?
  - Depends on what you want:
  - It helps **expose** ambiguity
  - ...but it does **not** remove it
  - parse ranking: choose one parse based on e.g. a language model





The Chomsky hierarchy

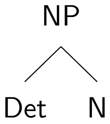


Context Free Grammars

- Context-Free Grammars generate Context-Free Languages
- CF languages fit into the Chomsky hierarchy between regular languages and context-sensitive languages
  - All regular languages are also context free languages
  - All sets of strings describable by FSAs can be described by a CFG
  - But not vice versa

Context Free Grammars

NP → Det N



- Represent constituent structure
- Encode a sharp notion of grammaticality
  - Compare to N-gram models

Grammaticality

- What is a **grammatical** sentence?
- What is an **ungrammatical** sentence?
- Which sentences are grammatical?
  - *I want to book a flight to Boston*
  - *I want to booked a flight to Boston*
  - *Colorless green ideas sleep furiously*
  - *Twas brillig, and the slithy toves did gyre and gimble in the wabe*

  - ...from a CFG's point of view?
  - ...from a probabilistic model point of view?
  - ...from a human point of view?

CFGs, informally

- Consist of **rules**, or **productions**
  - each expresses the ways that symbols of the language can be grouped and ordered
- ...and a **lexicon** of words and symbols.

NP → Det Nominal

NP → ProperNoun

Nominal → Noun | Nominal Noun

Det → *a*

Det → *the*

Noun → *flight*

CFGs, formally

- A CFG is a 4-tuple: $<$ C, $\sum$, P, S $>$:
    - C is the set of categories (aka non-terminals, e.g., { S, NP, VP, V, ...})
    - $\sum$ is the vocabulary (aka terminals, e.g., { Kim, snow, adores, ... })
    - P is the set of rewrite rules, of the form: $\alpha \rightarrow \beta_1, \beta_2, ..., \beta_n$
    - S (in C) is the start-symbol
    - For each rule $\alpha \rightarrow \beta_1, \beta_2, ..., \beta_n$ in P, α is drawn from C and each β is drawn from C or $\sum$

Parsing and Generation

- ...familiar dualism
    - recall FSTs
- **Parsing**: assigning a structure to a string
- **Generation**: using rules to write strings
- **Derivation**: Arriving from a string to a structure (or vice versa) by applying a series of rules

The Start symbol

- Is needed for us to know where to start (or finish), to get a well-formed structure
  - Would we want to start deriving from VP?
    - maybe! depends on the situation
- When denoted S, is easy to think of as referring to “sentence”, but it need not be the case
- It really refers to “Start”, not “sentence”.

CFG Example

- *Book my flight. Do you know the number? He gave me the number. He served my dinner.*
- Using the following lexicon, write rules that will generate (at least) these sentences, and assign them plausible structures.
  - Aux = {do, does}
  - V = {book, know, gave, serve, served}
  - N = {flight, number, dinner, you, me, he}
  - Det = {my, the}

CFG Example, candidate grammar

- *Book my flight. Do you know the number? He gave me the number. He served my dinner.*
    - Aux = {do, does}
    - V = {book, know, gave, serve, served}
    - N = {flight, number, dinner, you, me, he}
    - Det = {my, the}
    - S → NP VP | VP | Aux S
    - NP → Det N
    - VP → V NP (NP)

What is missing? How to fix it?

CFG Example, candidate grammar

- *Book my flight. Do you know the number? He gave me the number. He served my dinner.*
  - Aux = {do, does}
  - V = {book, know, gave, serve, served}
  - N = {flight, number, dinner}
  - PRO = {me, you, he}
  - Det = {my, the}
  - S → NP VP | VP | Aux S
  - NP → Det N | PRO
  - VP → V NP (NP)

Better?

CFG Example

- *Book my flight. Do you know the number? He gave me the number. He served my dinner.*
  - Aux = {do, does}
  - V = {book, know, gave, serve, served}
  - N = {flight, number, dinner}
  - PRO = {me, you, he}
  - Det = {my, the}
  - S → NP VP | VP | Aux S
  - NP → Det N | PRO
  - VP → V NP (NP)

**He serve my dinner* (Problem?)

CFG Example

- *Book my flight. Do you know the number? He gave me the number. He served my dinner.*
    - Aux = {do, does}
    - V = {book, know, gave, serve, served, **serves**}
    - N = {flight, number, dinner}
    - PRO = {me, you, he}
    - Det = {my, the}
    - S → NP VP | VP | Aux S
    - NP → Det N | PRO
    - VP → V NP (NP)

**Does this flight serves you dinner* (Problem?)

CFG Example

- *Book my flight. Do you know the number? He gave me the number. He served my dinner.*
  - AuxSG = {does}
  - AuxPL = {do}
  - V-PL = {book, know, gave, serve, served}
  - V-SG = {books, knows, gave, serves, served}
  - N-SG = {flight, number, dinner}
  - N-PL = {flights, numbers, dinners}
  - PRO-SG = {me, you, he}
  - PRO-PL = {you}
  - Det = {my, the}
  - S → NP VP | VP | Aux S
  - NP-SG → Det N-SG | PRO
  - NP-SG → Det N-PL | PRO
  - VP → V NP (NP)

Problem?

Limitations of CFGs

The cat chases the mouse vs. **The cat chase the mouse*

- Can we model agreement?
  - Sure. But the grammar will quickly become rather huge and inelegant!
    - ...will need duplicate rules whenever 3rd person singular and plural are involved
    - ...what about languages with lots of various inflections?
- what about relating passive and interrogative sentences to their declarative counterparts?
- how many subtypes of verbs will we need?
  - For each **subcategorization frame**, we will need to duplicate all the appropriate rules

What about heads?

- A **head** in syntactic theory is an item that is the most important in the phrase
- Essential for dependency parsing and probabilistic parsing
- Can we augment CFG with heads?

What about heads?

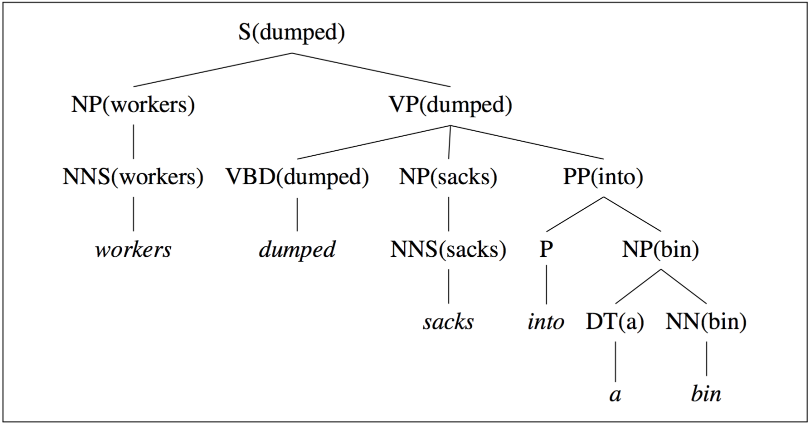


Figure 11.11 A lexicalized tree from Collins (1999).

Parsing algorithms and grammars

- A grammar is typically **input** to a parser
- (e.g. we've been parsing by hand, using CFGs)
- Grammars can be engineered or learned statistically from corpora
    - Both approaches have pros and cons
    - In particular, engineered grammars have higher **precision** while statistically-learned grammars have higher **recall**
    - why?

Evaluating parsing

- How would you do extrinsic evaluation of a parsing system?
- How would you do intrinsic evaluation?
  - Gold standard data?
  - Metrics?

Gold standard

- What would a gold standard look like?

Gold standard

- A corpus of string-to-structure mappings
- Is this different from a corpus of hand-written digit to actual digit mappings?
- From a corpus of string-to-POS sequence mappings?

Gold standard

- A corpus of string-to-structure mappings
- But: there's no ground truth in trees!
- Semantic dependencies might be easier to get cross-framework agreement on, but even there it's non-trivial
- The Penn Treebank (Marcus et al 1993) was originally conceived of as a target for cross-framework parser evaluation
- For project-internal/regression testing, grammar-based treebanking is effective for creating (g)old-standard data

Metrics: Parseval

- Labeled precision:

$$\frac{\text{\# of correct constituents in candidate parse}}{\text{total \# of constituents in candidate parse}}$$

- Labeled recall:

$$\frac{\text{\# of correct constituents in candidate parse}}{\text{total \# of constituents in gold standard parse}}$$

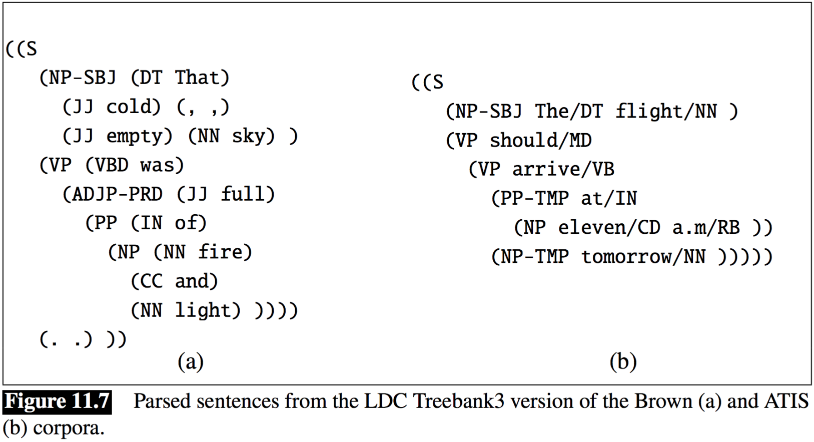
  - Constituents defined by starting point, ending point, and non-terminal symbol of spanning node

- Cross brackets: average number of constituents where the phrase boundaries of the gold standard and the candidate parse overlap

  - Example overlap: ((A B) C) v. (A (B C))

Treebanking

- A **treebank** is a syntactically annotated corpus



```
((S
   (NP-SBJ (DT That)
     (JJ cold) (, ,)
     (JJ empty) (NN sky) )
   (VP (VBD was)
     (ADJP-PRD (JJ full)
       (PP (IN of)
         (NP (NN fire)
           (CC and)
           (NN light) ))))
   (. .) ))
```

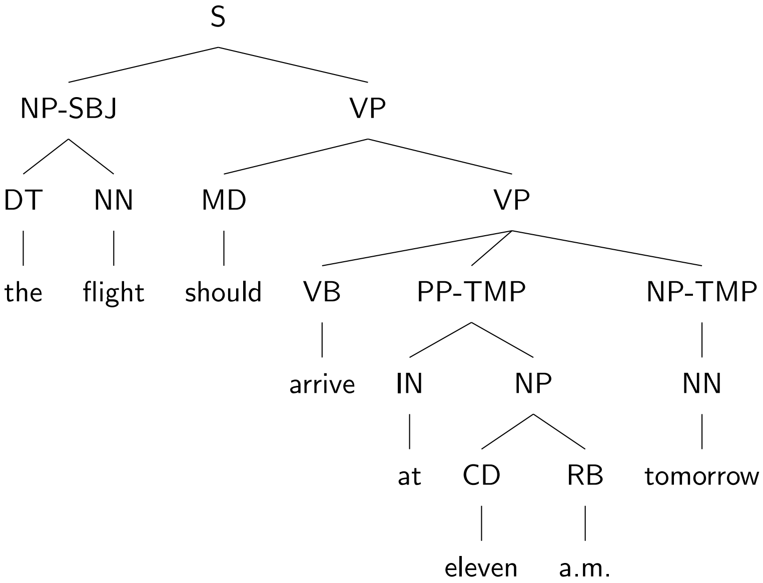
(a)

```
((S
   (NP-SBJ The/DT flight/NN )
   (VP should/MD
     (VP arrive/VB
       (PP-TMP at/IN
         (NP eleven/CD a.m/RB ))
       (NP-TMP tomorrow/NN )))))
```

(b)

Figure 11.7 Parsed sentences from the LDC Treebank3 version of the Brown (a) and ATIS (b) corpora.

Tree visualization



Treebanks as grammars

- How to turn a treebank into a grammar?
  - Extract the rewrite rules
  - We could also count how many times we saw which production
    - Anything useful we could do with that?

Treebanks as grammars

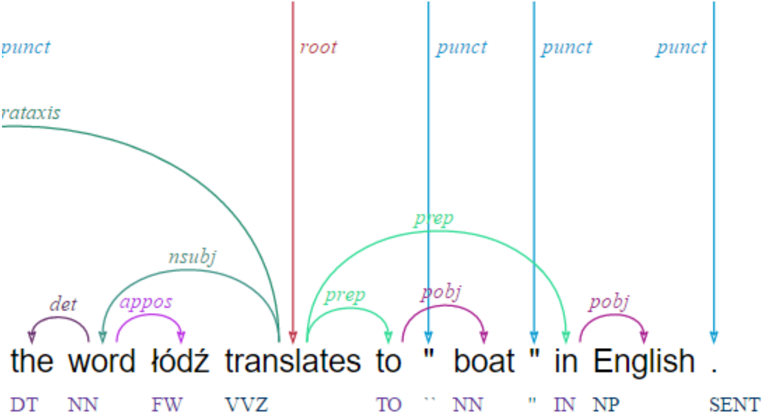
- How to turn a treebank into a grammar?
  - Extract the rewrite rules
  - Exercise: extract the rules from the sample treebank sentences in the previous slide

Treebanks as grammars

- How to turn a treebank into a grammar?
  - Extract the rewrite rules
  - We could also count how many times we saw which production (See `https://youtu.be/HSAUqdDnYEU`)
    - Anything useful we could do with that?
    - Statistical parsing!

Dependency Formalisms and Treebanks

- The Penn Treebank (Marcus et al., 1993)
- Universal Dependencies (Nivre et al., 2016)


punct
root
punct
punct
punct
rataxis
prep
nsubj
det
appos
prep
pobj
pobj
the word łódź translates to " boat " in English .
DT NN FW VVZ TO `` NN '' IN NP SENT

Dependency Treebank

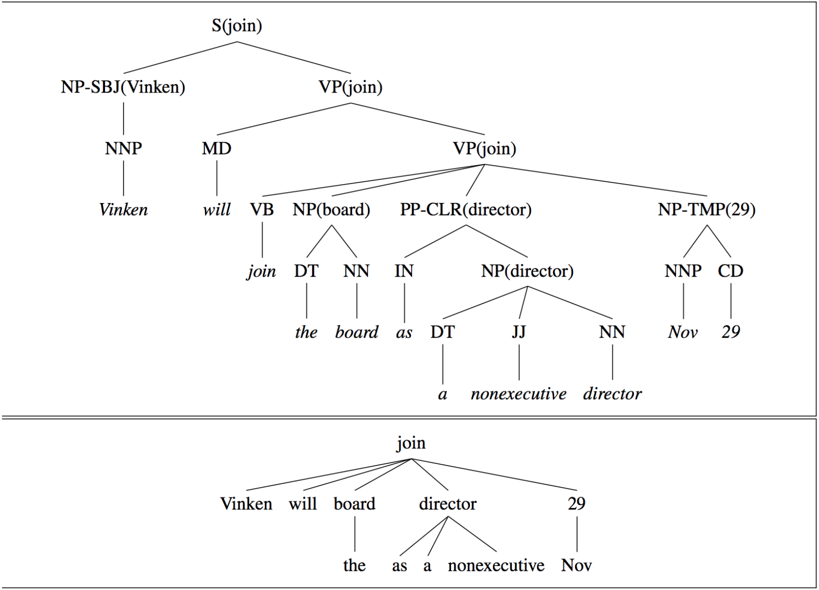


Figure 14.4 A phrase-structure tree from the *Wall Street Journal* component of the Penn Treebank 3.

What you need to know

- Parsing, grammar, grammaticality definitions
- Bracket and tree notation
- CFGs: informal definition, production, derivation
- Treebanks (what they are)