

Lecture 15: English Syntax & CFGs

Nathan Schneider

(most slides from Marine Carpuat)

ENLP | 19 March 2018

Today's Agenda

- From sequences to **trees**
- Syntax
 - Constituent, Grammatical relations, Dependency relations
- Formal Grammars
 - Context-free grammar
 - Dependency grammars
- Treebanks

sýntaxis (setting out or arranging)

- The ordering of words and how they group into phrases
 - [[the old man] [is yawning]]
 - [[the old] [man the boats]]

Syntax and Grammar

- Goal of syntactic theory
 - “explain how people combine words to form sentences and how children attain knowledge of sentence structure”
- Grammar
 - implicit knowledge of a native speaker
 - acquired without explicit instruction
 - minimally able to generate all and only the possible sentences of the language

[Philips, 2003]

Syntax vs. Meaning

“Colorless green ideas sleep furiously.”

— Noam Chomsky (1957)

You can tell that the words are in the right order.

- ...and that “colorless” and “green” modify “ideas”
- ...and that ideas sleep
- ...and that the sleeping is done furiously
- ...and that it sounds like an English sentence, even if you can't imagine what it means.
- Contrast with: “sleep green furiously ideas colorless”

But isn't meaning more important?

[send [the text message from James] [to Sharon]]

[translate [the message] [from Hindi] [to English]]

- When you say these to your phone, you want it to respond appropriately.
- We will see that syntax helps you find the meaning.

Syntax in NLP

- Syntactic analysis often a key component in applications
 - Grammar checkers
 - Dialogue systems
 - Question answering
 - Information extraction
 - Machine translation
 - ...

Two views of syntactic structure

- Constituency (phrase structure)
 - Phrase structure organizes words in nested constituents
- Dependency structure
 - Shows which words depend on (modify or are arguments of) which on other words

CONSTITUENCY PARSING & CONTEXT FREE GRAMMARS

Constituency

- Basic idea: groups of words act as a single unit
- Constituents form coherent classes that behave similarly
 - With respect to their internal structure: e.g., at the core of a noun phrase is a noun
 - With respect to other constituents: e.g., noun phrases generally occur before verbs

Constituency: Example

- The following are all noun phrases in English...

Harry the Horse

the Broadway coppers

they

a high-class spot such as Mindy's

the reason he comes into the Hot Box

three parties from Brooklyn

- Why?
 - They can all precede verbs
 - They can all be preposed/postposed
 - ...

Grammars and Constituency

- For a particular language:
 - What are the “right” set of constituents?
 - What rules govern how they combine?
- Answer: not obvious and difficult
 - That’s why there are many different theories of grammar and competing analyses of the same data!
- Our approach
 - Focus primarily on the “machinery”

Finite-State/Regular Grammars

- You've already seen one class of grammars: **regular expressions**
 - A pattern like `^[a-z][0-9]$` corresponds to a grammar which **accepts** (matches) some strings but not others.
 - Can regular languages define *infinite* languages?
 - Can regular languages define *arbitrarily complex* languages?

Finite-State/Regular Grammars

- You've already seen one class of grammars: **regular expressions**
 - A pattern like `^[a-z][0-9]$` corresponds to a grammar which **accepts** (matches) some strings but not others.
 - Can regular languages define *infinite* languages? **Yes, e.g.: a^***
 - Can regular languages define *arbitrarily complex* languages? **No. Cannot match all strings with matched parentheses (recursion/arbitrary nesting).**

Context-Free Grammars

- Context-free grammars (CFGs)
 - Aka phrase structure grammars
 - Aka Backus-Naur form (BNF)
- Consist of
 - Rules
 - Terminals
 - Non-terminals

Context-Free Grammars

- Terminals
 - We'll take these to be words (for now)
- Non-Terminals
 - The constituents in a language (e.g., noun phrase)
- Rules
 - Consist of a single non-terminal on the left and any number of terminals and non-terminals on the right

An Example Grammar

Grammar Rules	Examples
$S \rightarrow NP VP$	I + want a morning flight
$NP \rightarrow$ <i>Pronoun</i> <i>Proper-Noun</i> <i>Det Nominal</i>	I Los Angeles a + flight
$Nominal \rightarrow$ <i>Nominal Noun</i> <i>Noun</i>	morning + flight flights
$VP \rightarrow$ <i>Verb</i> <i>Verb NP</i> <i>Verb NP PP</i> <i>Verb PP</i>	do want + a flight leave + Boston + in the morning leaving + on Thursday
$PP \rightarrow$ <i>Preposition NP</i>	from + Los Angeles

CFG: Formal definition

N a set of **non-terminal symbols** (or **variables**)

Σ a set of **terminal symbols** (disjoint from N)

R a set of **rules** or productions, each of the form $A \rightarrow \beta$,
where A is a non-terminal,

β is a string of symbols from the infinite set of strings $(\Sigma \cup N)^*$

S a designated **start symbol**

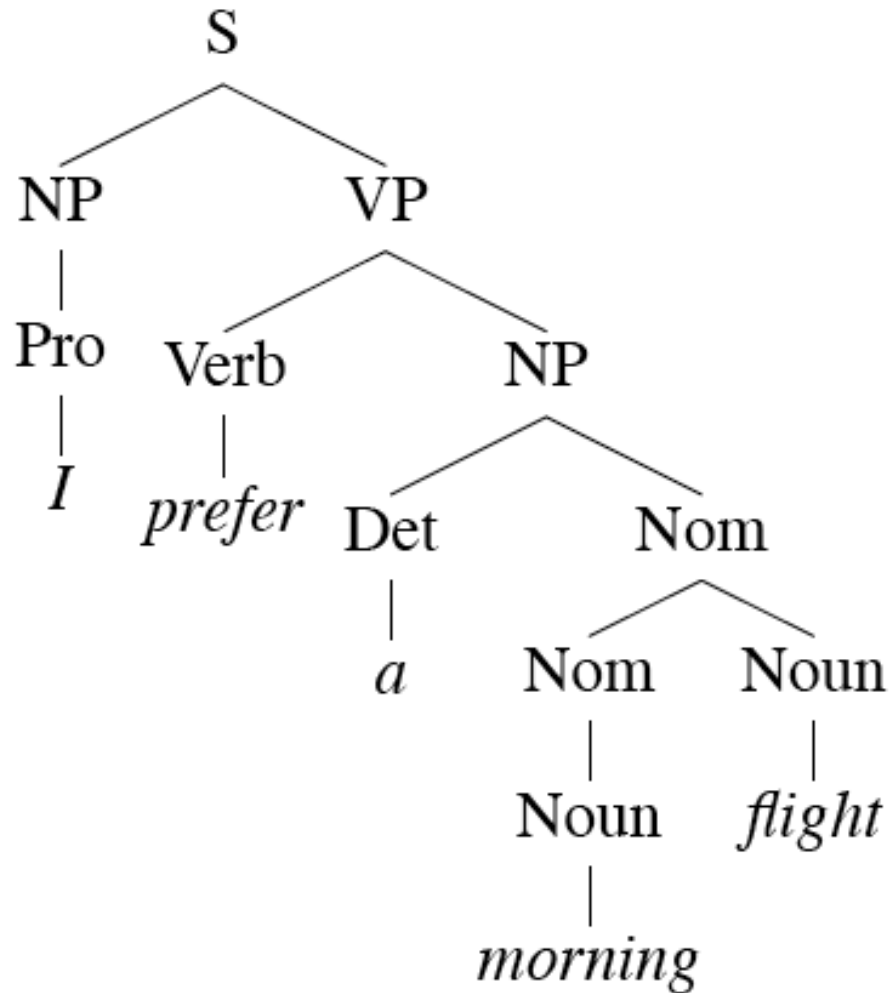
Three-fold View of CFGs

- Generator
- Acceptor
- Parser

Derivations and Parsing

- A **derivation** is a sequence of rules applications that
 - Covers all tokens in the input string
 - Covers only the tokens in the input string
- **Parsing**: given a string and a grammar, recover the derivation
 - Derivation can be represented as a parse tree
 - Multiple derivations?

Parse Tree: Example



An English Grammar Fragment

- Sentences
- Noun phrases
 - Issue: agreement
- Verb phrases
 - Issue: subcategorization

Sentence Types

- Declaratives: A plane left.
 $S \rightarrow NP VP$
- Imperatives: Leave!
 $S \rightarrow VP$
- Yes-No Questions: Did the plane leave?
 $S \rightarrow Aux NP VP$
- WH Questions: When did the plane leave?
 $S \rightarrow WH-NP Aux NP VP$

Noun Phrases

- We have seen rules such as

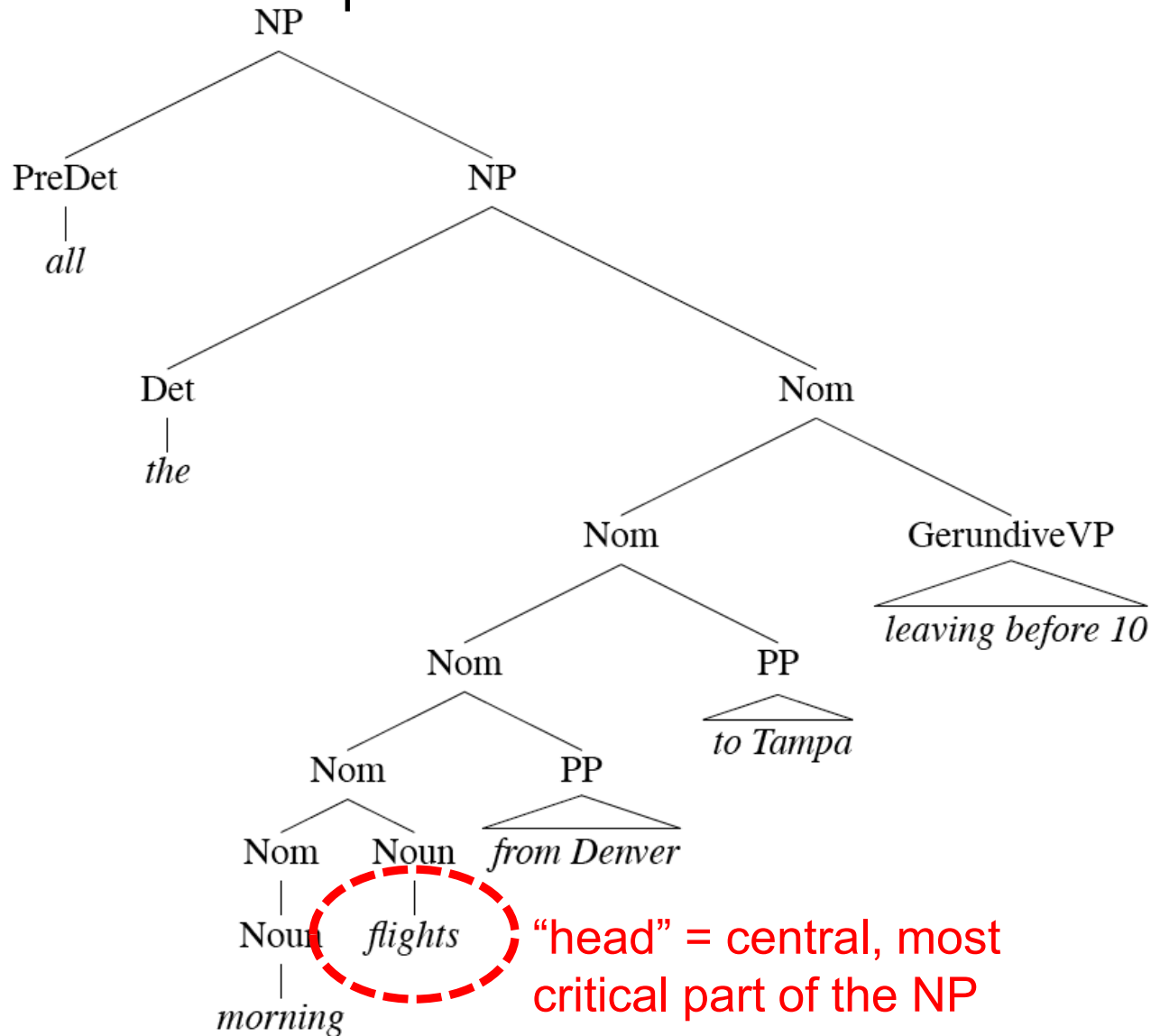
NP → *Det Nominal*

NP → *ProperNoun*

Nominal → *Noun* | *Nominal Noun*

- But NPs are a bit more complex than that!
 - E.g. “All the morning flights from Denver to Tampa leaving before 10”

A Complex Noun Phrase



Determiners

- Noun phrases can start with determiners...
- Determiners can be
 - Simple lexical items: the, this, a, an, etc. (e.g., "a car")
 - Or simple possessives (e.g., "John's car")
 - Or complex recursive versions thereof (e.g., John's sister's husband's son's car)

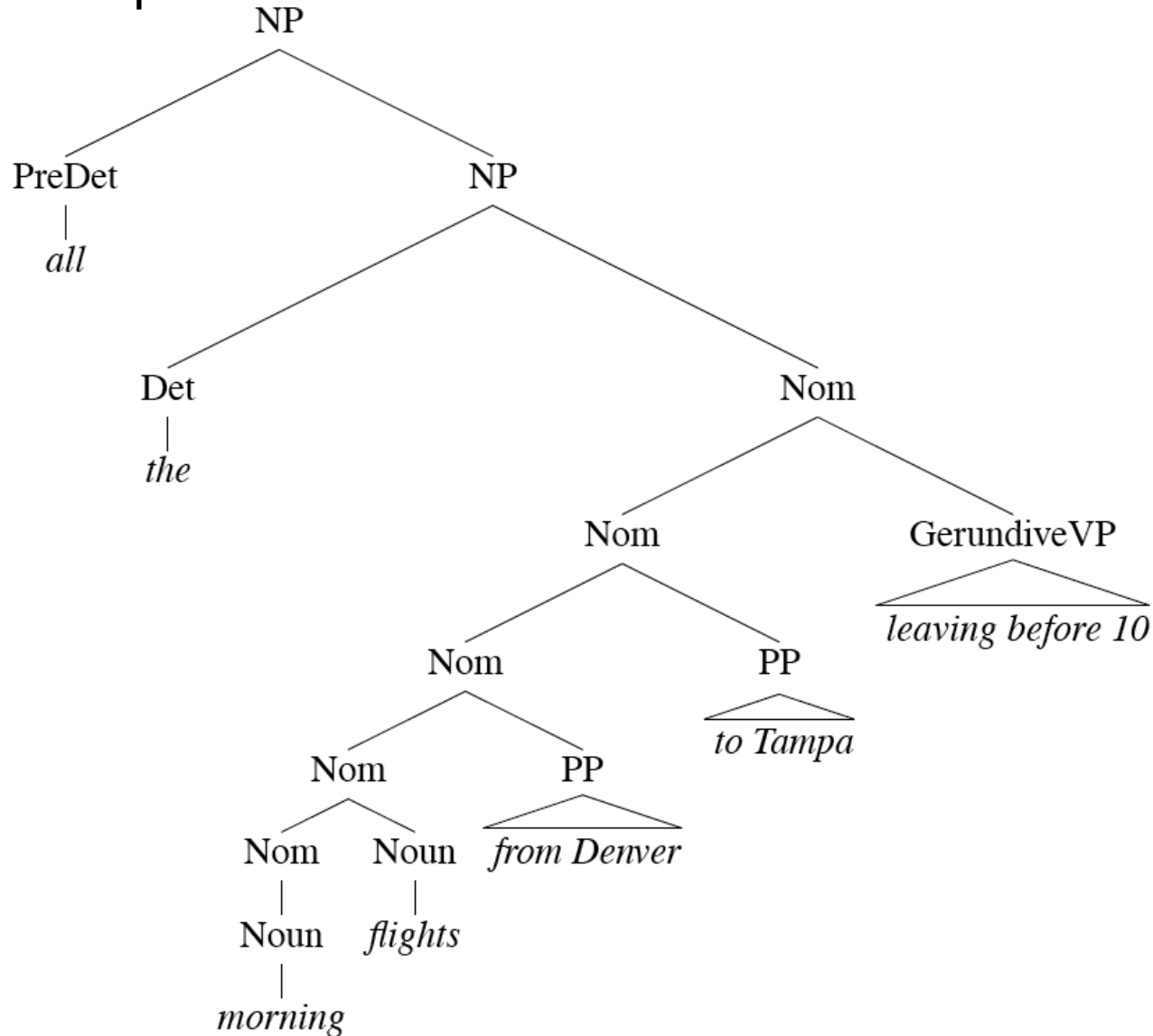
Premodifiers

- Come before the head
- Examples:
 - Cardinals, ordinals, etc. (e.g., "three cars")
 - Adjectives (e.g., "large car")
- Ordering constraints
 - "three large cars" vs. "?large three cars"

Postmodifiers

- Come after the head
- Three kinds
 - Prepositional phrases (e.g., "from Seattle")
 - Non-finite clauses (e.g., "arriving before noon")
 - Relative clauses (e.g., "that serve breakfast")
- Similar recursive rules to handle these
 - Nominal → Nominal PP
 - Nominal → Nominal GerundVP
 - Nominal → Nominal RelClause

A Complex Noun Phrase Revisited



Subject and Object

Syntactic (not semantic):

The batter hit the ball [subject is semantic *agent*]

The ball was hit by the batter [subject is semantic *patient*]

The ball was given a whack by the batter
[subject is semantic *recipient*]

{George, the key, the wind} opened the door

Subject ≠ topic:

I just married the most beautiful woman in the world

Now **beans**, I like

As for democracy, I think it's the best form of government

Subject and Object

- English subjects

- agree with the verb
- when pronouns, in **nominative** case (I/she/he/we/they)
- omitted from infinitive clauses (I tried _ to read the book, I hoped _ to be chosen)

- English objects

- when pronouns, in **accusative** case (me/her/him/us/them)
- become subjects in passive sentences

Agreement

- Agreement: constraints that hold among various constituents
- Example, number agreement in English

This flight

Those flights

One flight

Two flights

*This flights

*Those flight

*One flights

*Two flight

Problem

- Our NP rules don't capture agreement constraints
 - Accepts grammatical examples (this flight)
 - Also accepts ungrammatical examples (*these flight)
- Such rules **overgenerate**

Possible CFG Solution

- Encode agreement in non-terminals:
 - SgS \rightarrow SgNP SgVP
 - PIS \rightarrow PINP PIVP
 - SgNP \rightarrow SgDet SgNom
 - PINP \rightarrow PIDet PINom
 - PIVP \rightarrow PIV NP
 - SgVP \rightarrow SgV Np

Verb Phrases

- English verb phrases consists of
 - Head verb
 - Zero or more following constituents (called arguments)
- Sample rules:

VP → *Verb* disappear

VP → *Verb NP* prefer a morning flight

VP → *Verb NP PP* leave Boston in the morning

VP → *Verb PP* leaving on Thursday

Subcategorization

- Not all verbs are allowed to participate in all VP rules
 - We can subcategorize verbs according to argument patterns (sometimes called “frames”)
 - Modern grammars may have 100s of such classes

Subcategorization

- Sneeze: John sneezed
- Find: Please find [a flight to NY]_{NP}
- Give: Give [me]_{NP} [a cheaper fare]_{NP}
- Help: Can you help [me]_{NP} [with a flight]_{PP}
- Prefer: I prefer [to leave earlier]_{TO-VP}
- Told: I was told [United has a flight]_S
- ...

Subcategorization

- Subcategorization at work:
 - *John sneezed the book
 - *I prefer United has a flight
 - *Give with a flight
- But some verbs can participate in multiple frames:
 - I ate
 - I ate the apple
- How do we formally encode these constraints?

Why?

- As presented, the various rules for VPs overgenerate:

VP → *Verb* disappear

VP → *Verb NP* prefer a morning flight

VP → *Verb NP PP* leave Boston in the morning

VP → *Verb PP* leaving on Thursday

- John sneezed [the book]_{NP}
 - Allowed by the second rule...

Possible CFG Solution

- Encode agreement in non-terminals:
 - SgS \rightarrow SgNP SgVP
 - PIS \rightarrow PINP PIVP
 - SgNP \rightarrow SgDet SgNom
 - PINP \rightarrow PIDet PINom
 - PIVP \rightarrow PIV NP
 - SgVP \rightarrow SgV Np
- Can use the same trick for verb subcategorization

Grammar Formalisms

- Linguists have invented grammar formalisms that overcome the limitations of Context-Free Grammars
 - Lexical Functional Grammar
 - Head-Driven Phrase Structure Grammar
 - Combinatory Categorical Grammar
 - Lexicalized Tree-Adjoining Grammar
 - Grammatical Framework
- We sometimes teach a class on these.

Recap: Three-fold View of CFGs

- Generator
- Acceptor
- Parser

Recap: why use CFGs in NLP?

- CFGs have about just the right amount of machinery to account for basic syntactic structure in English
 - Lot's of issues though...
- Good enough for many applications!
 - But there are many alternatives out there...

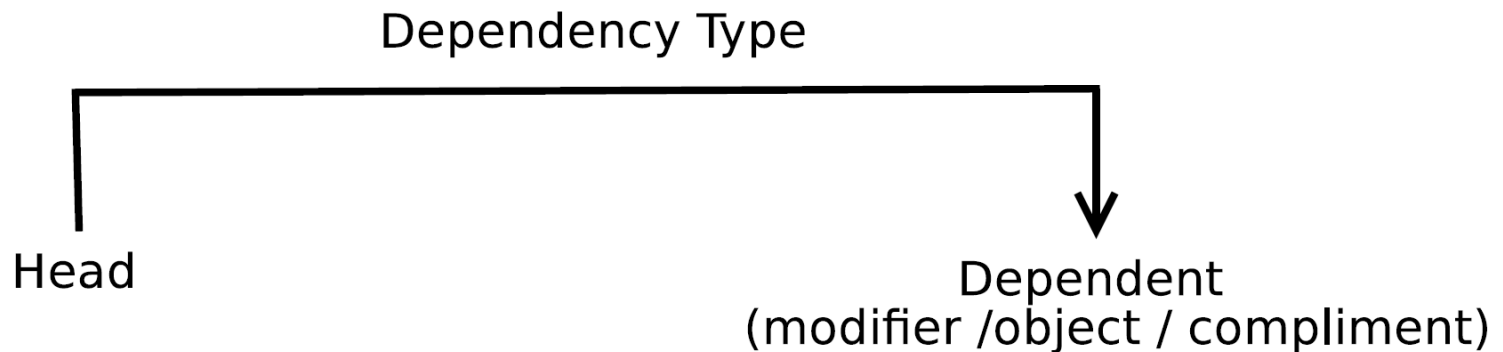
DEPENDENCY GRAMMARS

Dependency Grammars

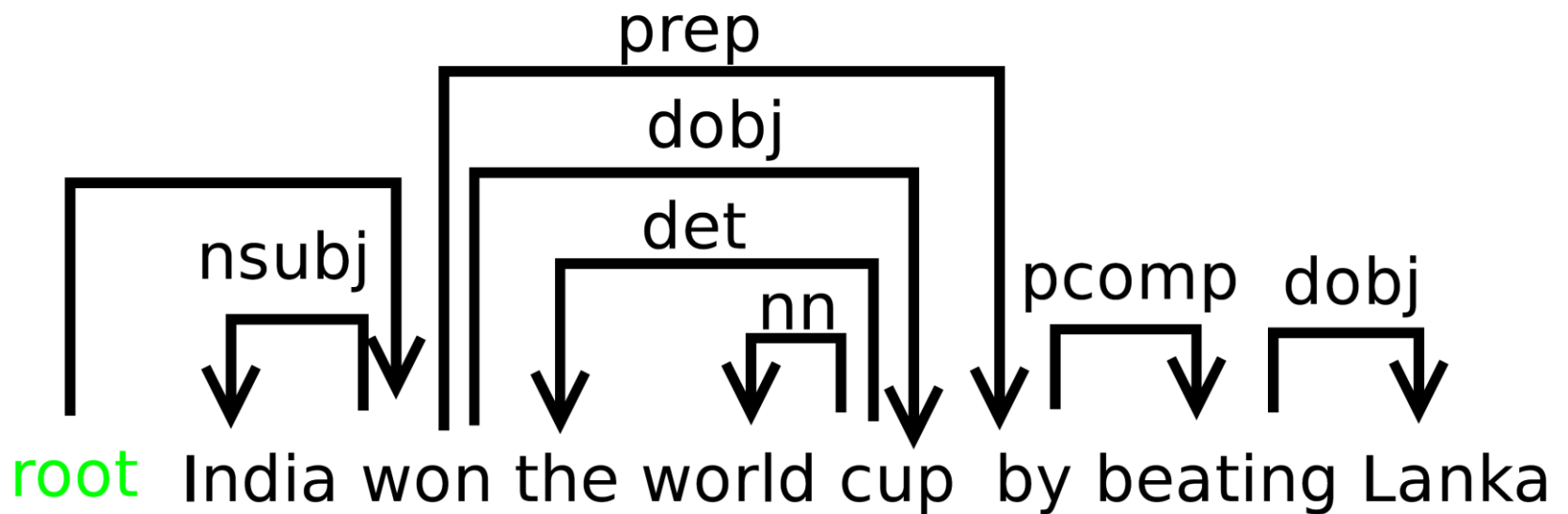
- CFGs focus on constituents
 - Non-terminals don't actually appear in the sentence
- In dependency grammar, a parse is a graph (usually a tree) where:
 - Nodes represent words
 - Edges represent dependency relations between words (typed or untyped, directed or undirected)

Dependency Grammars

- Syntactic structure = lexical items linked by binary asymmetrical relations called dependencies



Example Dependency Parse



TREEBANKS

Treebanks

- Treebanks are corpora in which each sentence has been paired with a parse tree
- These are generally created:
 - By first parsing the collection with an automatic parser
 - And then having human annotators correct each parse as necessary
- But
 - Detailed annotation guidelines are needed
 - Explicit instructions for dealing with particular constructions

Penn Treebank

- Penn TreeBank is a widely used treebank
 - 1 million words from the Wall Street Journal
- Treebanks implicitly define a grammar for the language

Penn Treebank: Example

```
( (S ( ' ' ' ' )
  (S-TPC-2
    (NP-SBJ-1 (PRP We) )
    (VP (MD would)
      (VP (VB have)
        (S
          (NP-SBJ (-NONE- *-1) )
          (VP (TO to)
            (VP (VB wait)
              (SBAR-TMP (IN until)
                (S
                  (NP-SBJ (PRP we) )
                  (VP (VBP have)
                    (VP (VBN collected)
                      (PP-CLR (IN on)
                        (NP (DT those)(NNS assets))))))))))))))
    ( , , ) ( ' ' ' ' )
    (NP-SBJ (PRP he) )
    (VP (VBD said)
      (S (-NONE- *T*-2) ))
    ( . . ) ))
```

Treebank Grammars

- Such grammars tend to be very flat
 - Recursion avoided to ease annotators burden
- Penn Treebank has 4500 different rules for VPs, including...
 - $VP \rightarrow VBD PP$
 - $VP \rightarrow VBD PP PP$
 - $VP \rightarrow VBD PP PP PP$
 - $VP \rightarrow VBD PP PP PP PP$

Summary

- Syntax & Grammar
- Two views of syntactic structures
 - Context-Free Grammars
 - Dependency grammars
 - Can be used to capture various facts about the structure of language (but not all!)
- Treebanks as an important resource for NLP