Syntax: Constituency and Dependency Structure

CS 490A, Fall 2021

https://people.cs.umass.edu/~brenocon/cs490a_f21/

Laure Thompson and Brendan O'Connor

College of Information and Computer Sciences

University of Massachusetts Amherst

Announcements

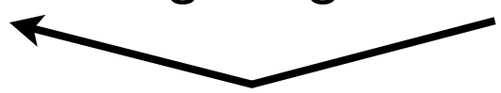
- HW3 coming out shortly (show)
  - Due Friday, Oct 29
  - Work with POS tags, NER, dependency parsing

- Project proposals due tomorrow!
  - Question: how will they be graded?

- Discuss project ideas after class?
  - (outdoors please)

- Syntax: how do words structurally combine to form sentences and meaning?

- Representations
  - Constituents
    - [the big dogs] chase cats
    - [colorless green clouds] chase cats
  - Dependencies
    - The **dog** ← **chased** the cat.
    - My **dog**, who's getting old, **chased** the cat.



- Idea of a *grammar (G)*: global template for how sentences / utterances / phrases ***w*** are formed, via latent syntactic structure ***y***
  - Linguistics: what do G and P(w,y | G) look like?
  - Generation: score with, or sample from, P(w, y | G)
  - Parsing: predict P(y | w, G)

Syntax for NLP

- If we could predict syntactic structure from raw text, that could help with...
  - Language understanding: meaning formed from structure
  - Grammar checking
  - Preprocessing: Extract phrases and semantic relationships between words for features, viewing, etc.
- Provides a connection between the theory of *generative linguistics* and computational modeling of language

Is language context-free?

- Regular language: repetition of repeated structures
  - e.g. "base noun phrases": (Noun | Adj)* Noun
    - subset of the JK pattern
- Context-free: hierarchical recursion
- Center-embedding: classic theoretical argument for CFG vs. regular languages
  - (10.1) The cat is fat.
  - (10.2) The cat that the dog chased is fat.
  - (10.3) *The cat that the dog is fat.
  - (10.4) The cat that the dog that the monkey kissed chased is fat.
  - (10.5) *The cat that the dog that the monkey chased is fat.

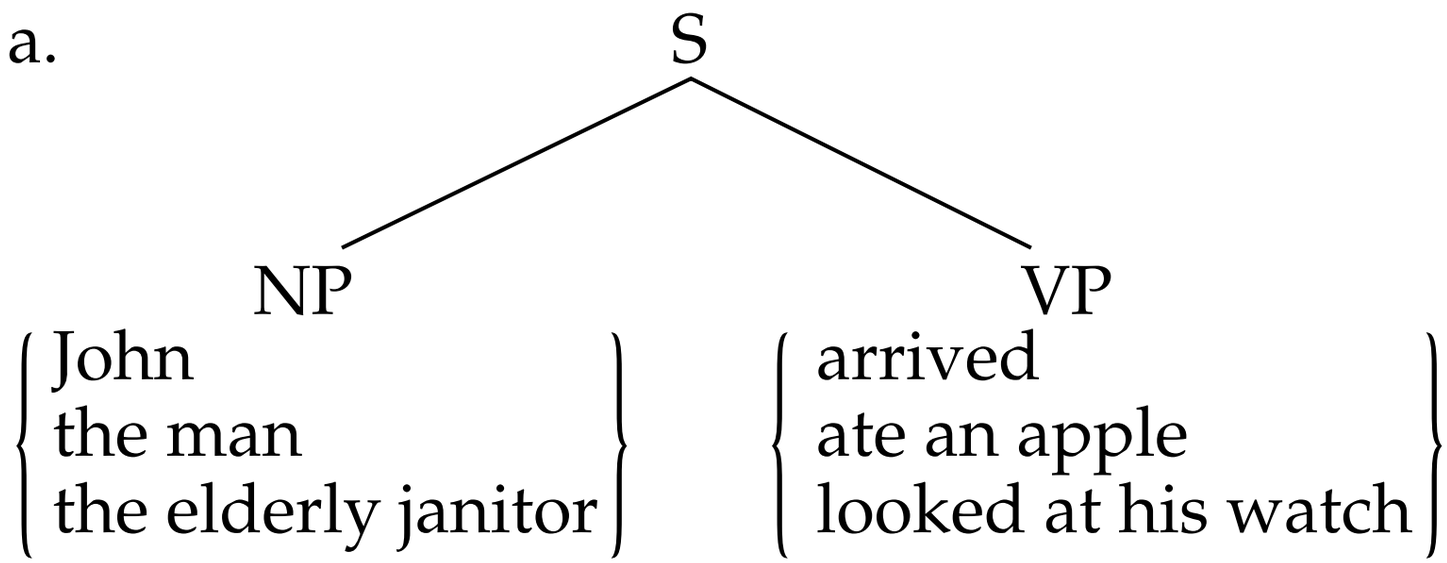
- Competence vs. Performance?

[Examples from Eisenstein (2017)]

Hierarchical view of syntax

- "a Sentence made of Noun Phrase followed by a Verb Phrase"

a.



b. S → NP VP (1)

[From Phillips (2003)]

Context-free grammars (CFG)

- A CFG is a 4-tuple:

N	a set of non-terminals
Σ	a set of terminals (distinct from N)
R	a set of productions, each of the form $A \rightarrow \beta$, where $A \in N$ and $\beta \in (\Sigma \cup N)^*$
S	a designated start symbol

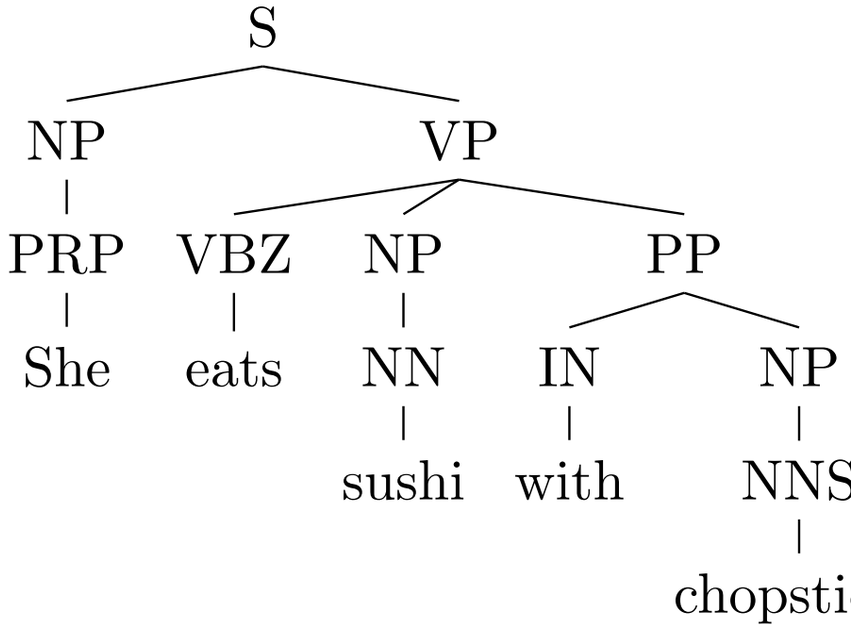
Example: see handout!

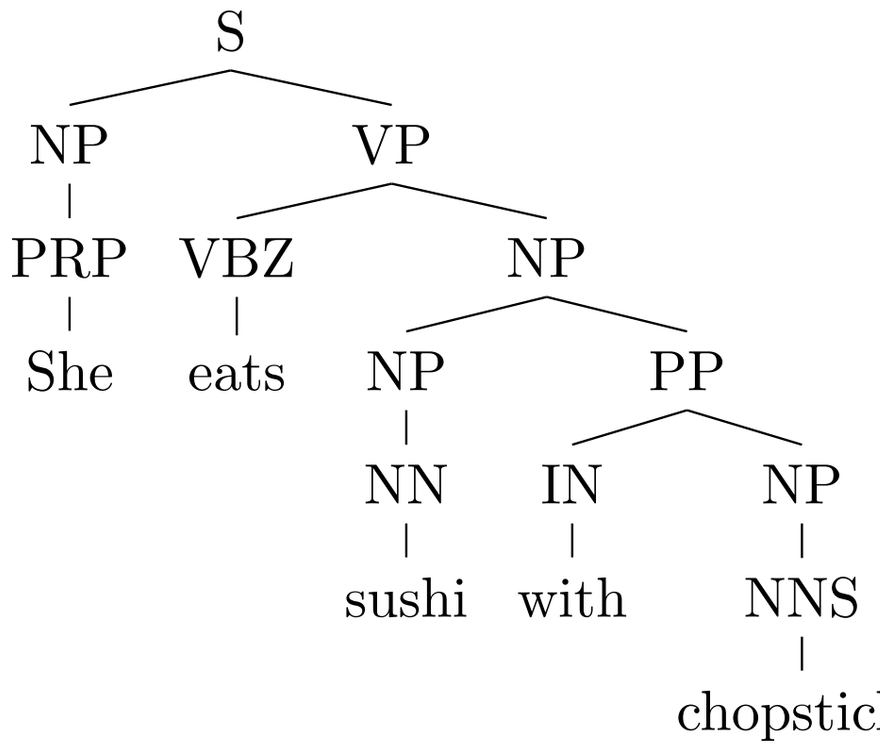
- Derivation: a sequence of rewrite steps from S to a string (sequence of terminals, i.e. words)
- Yield: the final string (sentence)
- The parse tree or constituency tree corresponds to the rewrite steps that were used to derive the string

- A CFG is a "boolean language model"
- A probabilistic CFG is a probabilistic language model:
  - Every production rule has a probability; defines prob dist. over strings.

- Example: derivation from worksheet's grammar

Example





(S(NP(PRP *She*)(VP(VBZ *eats*)
(NP(NN *sushi*))
(PP(IN*with*)(NP(NNS *chopsticks*))))))

(S(NP(PRP *She*)(VP(VBZ *eats*)
(NP(NP(NN *sushi*))(PP (IN*with*)(NP(NNS *chopsticks*)))))))

- All useful grammars are *ambiguous*: multiple derivations with same yield
- [Parse tree representations: Nested parens *or* non-terminal spans]

[Examples from Eisenstein (2017)]

Is language context-free?

- CFGs nicely explain nesting and agreement (if you stuff grammatical features into the non-terminals)
  - *The **processor** **has** I0 million times fewer transistors on it than todays typical micro-processors, **runs** much more slowly, and **operates** at five times the voltage...*

- $S \rightarrow NN\ VP$
  $VP \rightarrow VP3S \mid VPN3S \mid \ldots$
  $VP3S \rightarrow VP3S, VP3S, \text{and } VP3S \mid VBZ \mid VBZ\ NP \mid \ldots$

[Examples from Eisenstein (2017)]

- Ambiguities in syntax

Attachment ambiguity *we eat sushi with chopsticks, I shot an elephant in my pajamas.*

Modifier scope *southern food store*

Particle versus preposition *The puppy tore up the staircase.*

Complement structure *The tourists objected to the guide that they couldn't hear.*

Coordination scope *"I see," said the blind man, as he picked up the hammer and saw.*

Multiple gap constructions *The chicken is ready to eat*

[Examples from Eisenstein (2017)]

Probabilistic CFGs

$S \rightarrow NP\ VP$	[.80]	$Det \rightarrow that$ [.10] | a [.30] | the [.60]
$S \rightarrow Aux\ NP\ VP$	[.15]	$Noun \rightarrow book$ [.10] | $flight$ [.30]
$S \rightarrow VP$	[.05]	| $meal$ [.15] | $money$ [.05]
$NP \rightarrow Pronoun$	[.35]	| $flights$ [.40] | $dinner$ [.10]
$NP \rightarrow Proper\text{-}Noun$	[.30]	$Verb \rightarrow book$ [.30] | $include$ [.30]
$NP \rightarrow Det\ Nominal$	[.20]	| $prefer$; [.40]
$NP \rightarrow Nominal$	[.15]	$Pronoun \rightarrow I$ [.40] | she [.05]
$Nominal \rightarrow Noun$	[.75]	| me [.15] | you [.40]
$Nominal \rightarrow Nominal\ Noun$	[.20]	$Proper\text{-}Noun \rightarrow Houston$ [.60]
$Nominal \rightarrow Nominal\ PP$	[.05]	| TWA [.40]
$VP \rightarrow Verb$	[.35]	$Aux \rightarrow does$ [.60] | can [40]
$VP \rightarrow Verb\ NP$	[.20]	$Preposition \rightarrow from$ [.30] | to [.30]
$VP \rightarrow Verb\ NP\ PP$	[.10]	| on [.20] | $near$ [.15]
$VP \rightarrow Verb\ PP$	[.15]	| $through$ [.05]
$VP \rightarrow Verb\ NP\ NP$	[.05]	
$VP \rightarrow VP\ PP$	[.15]	
$PP \rightarrow Preposition\ NP$	[1.0]	

- Defines a probabilistic generative process for words in a sentence
  - Can parse with a modified form of CKY
- How to learn? Fully supervised with a treebank... unsupervised learning possible too, but doesn't give great results...

[J&M textbook]

Penn Treebank

```
( (S
    (NP-SBJ (NNP General) (NNP Electric) (NNP Co.) )
    (VP (VBD said)
      (SBAR (-NONE- 0)
        (S
          (NP-SBJ (PRP it) )
          (VP (VBD signed)
            (NP
              (NP (DT a) (NN contract) )
              (PP (-NONE- *ICH*-3) ))
            (PP (IN with)
              (NP
                (NP (DT the) (NNS developers) )
                (PP (IN of)
                  (NP (DT the) (NNP Ocean) (NNP State) (NNP Power) (NN project) ))))
            (PP-3 (IN for)
              (NP
                (NP (DT the) (JJ second) (NN phase) )
                (PP (IN of)
```

```
(NP
  (NP (DT an) (JJ independent)
    (ADJP
      (QP ($ $) (CD 400) (CD million) )
      (-NONE- *U*) )
    (NN power) (NN plant) )
  (, ,)
  (SBAR
    (WHNP-2 (WDT which) )
    (S
      (NP-SBJ-1 (-NONE- *T*-2) )
      (VP (VBZ is)
        (VP (VBG being)
          (VP (VBN built)
            (NP (-NONE- *-1) )
            (PP-LOC (IN in)
              (NP
                (NP (NNP Burrillville) )
                (, ,)
                (NP (NNP R.I) ))))))))))))))))
```

- A **dependency parse** is a tree or directed graph among the words in the sentence
- Directly encodes word-to-word grammatical relationships. Each edge is between:
  - **Head** (or **governor**), to the
  - **Dependent** (or **child**, or modifier)

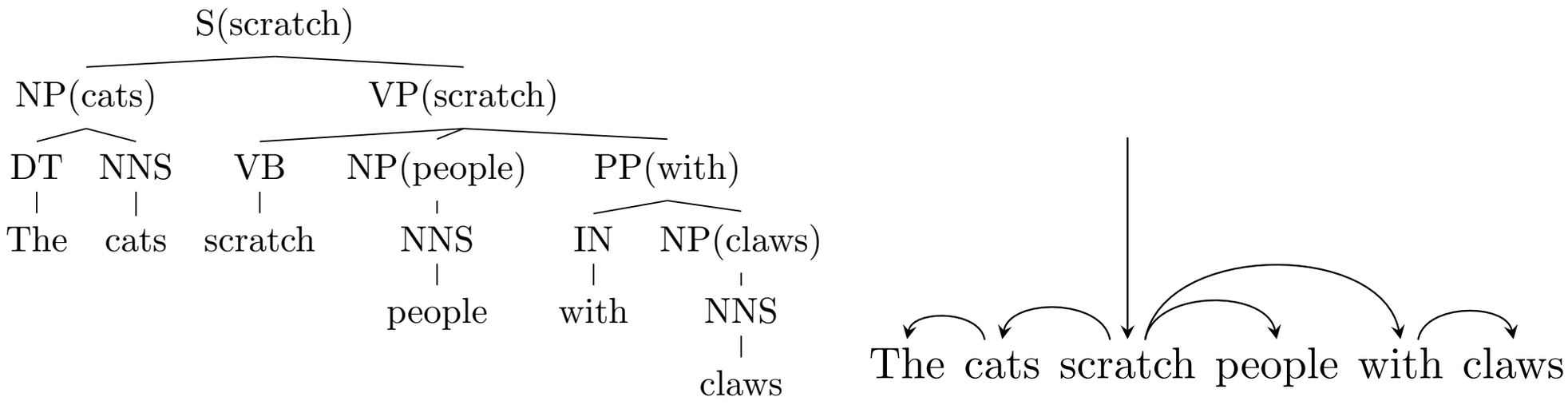
- [Example]

- There isn't really a generative grammar view of dependencies
  - They're more of a *descriptive* formalism
- Dependency parsers are often convenient to use for downstream applications
  - Subtree = a phrase
  - "Dependency bigrams": (parent,child) pairs in tree
    - Do these correspond to phrases?
- Dependency treebanks are available for many languages (https://universaldependencies.org/), and therefore parsers are widely available.

From Constituents to Dependencies

- Theories of grammar postulate that every phrase has a **head word**, which contains or typifies the grammatical content of the phrase

From Constituents to Dependencies



(a) lexicalized constituency parse

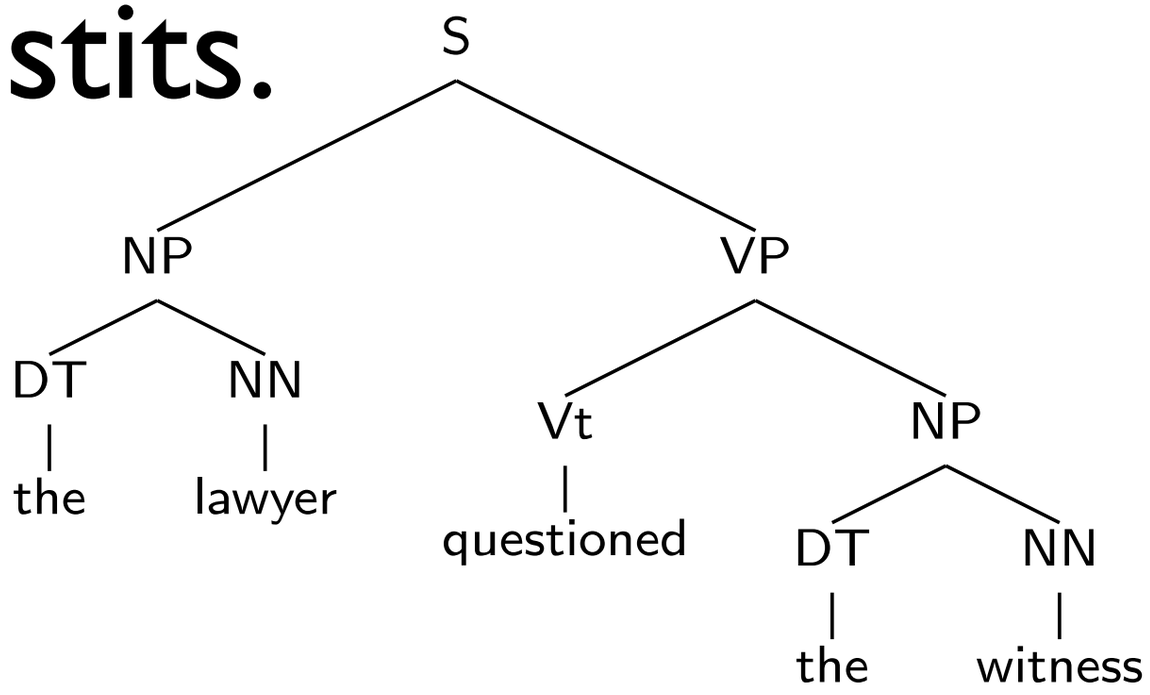
(b) unlabeled dependency tree

Figure 11.1: Dependency grammar is closely linked to lexicalized context free grammars: each lexical head has a dependency path to every other word in the constituent. (This example is based on the lexicalization rules from § 10.5.2, which make the preposition the head of a prepositional phrase. In the more contemporary Universal Dependencies annotations, the head of *with claws* would be *claws*, so there would be an edge *scratch* → *claws*.)

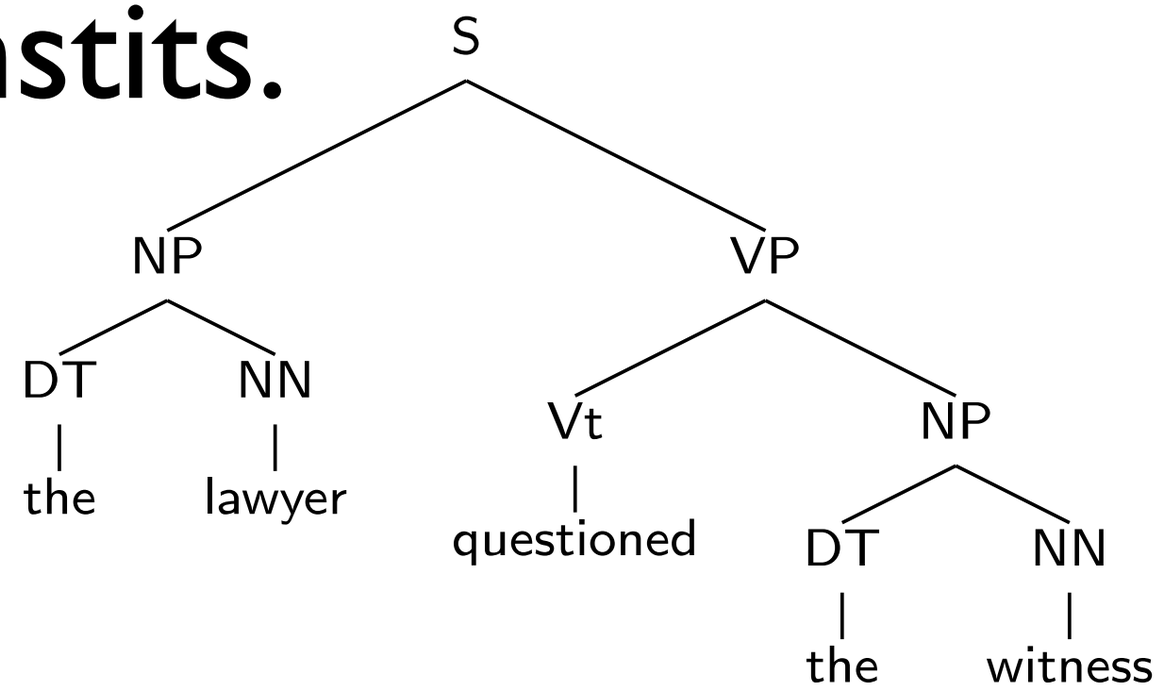
[Eisenstein (2017)]

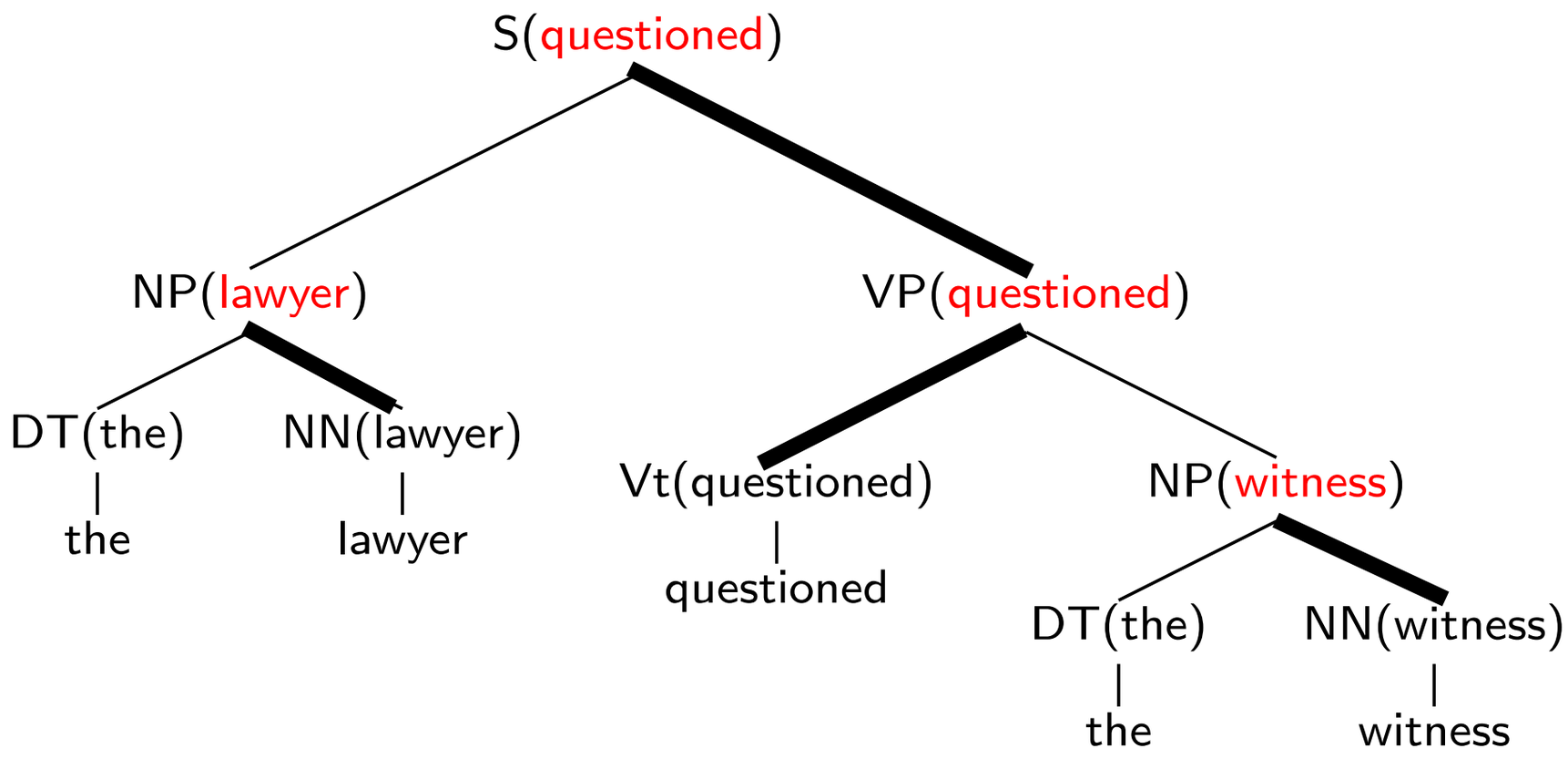
- stopped here 10/19

Heads in constits.



Heads in constits.





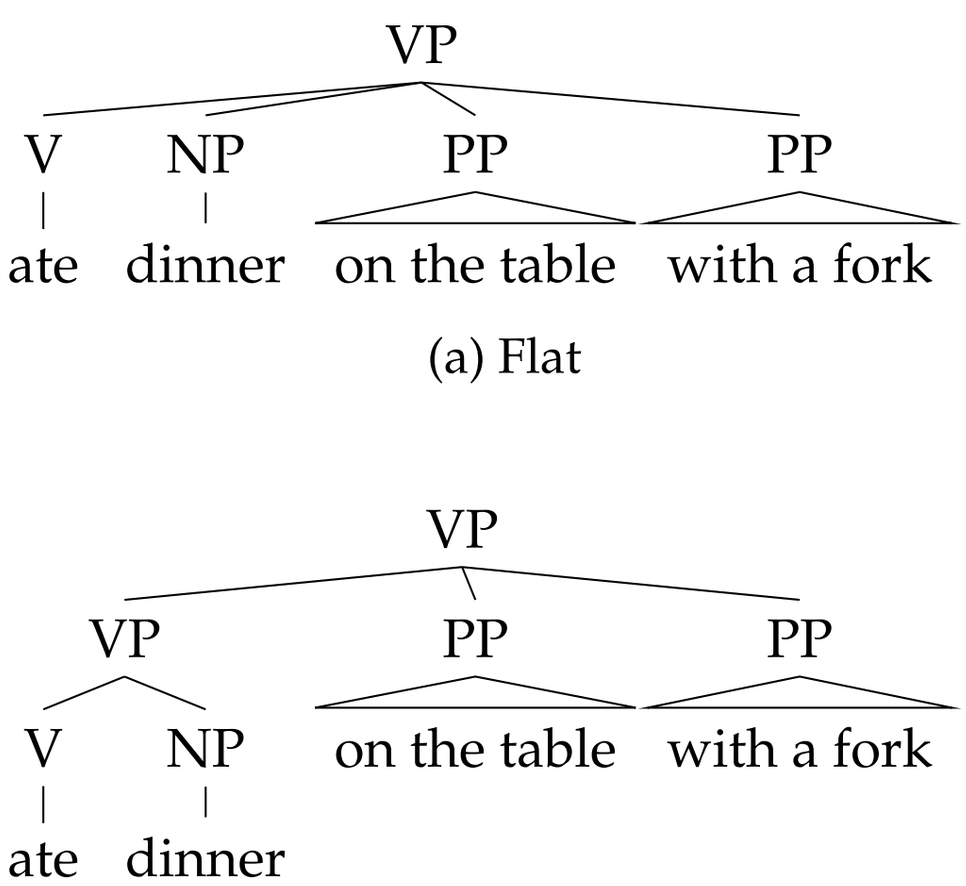
Head rules

- Idea: Every phrase has a head word
- Head rules: for every nonterminal in tree, choose one of its children to be its "head". This will define head words.
- Every nonterminal type has a different head rule; e.g. from Collins (1997):

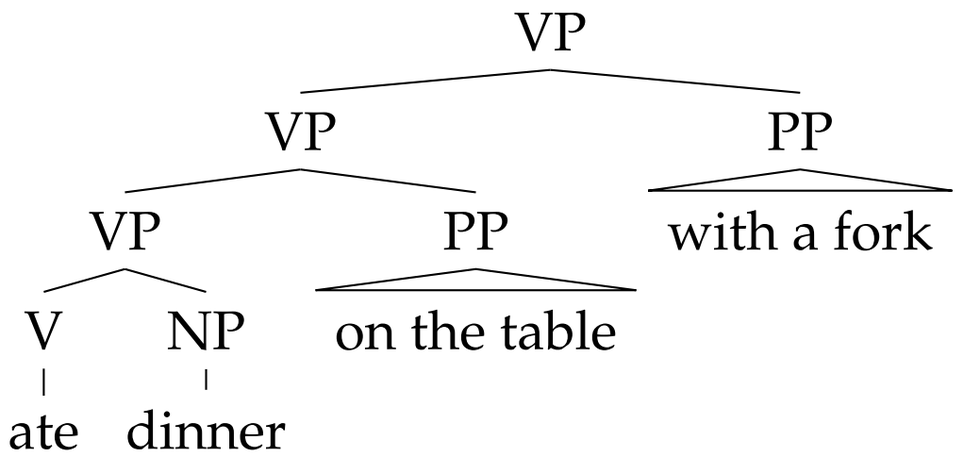
- If parent is NP,
  - Search from right-to-left for first child that's NN, NNP, NNPS, NNS, NX, JJR
  - Else: search left-to-right for first child which is NP

- Heads are super useful if you want just a single token to stand-in for a phrase!
  - Not just dep parsing. Entities, etc.

- Dependencies tend to be less specific than constituent structure

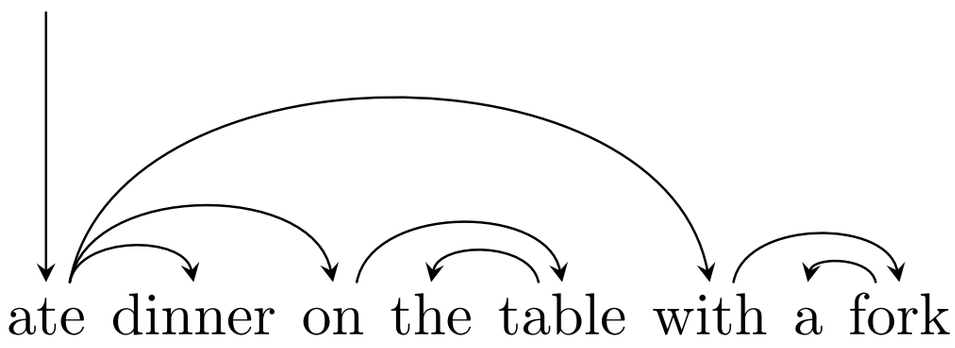


(a) Flat



(b) Two-level (PTB-style)

(c) Chomsky adjunction



(d) Dependency representation

[Eisenstein (2017)]