CS 489/698: Introduction to Natural Language Processing

Lecture 7: Syntax

Instructor: Freda Shi

fhs@uwaterloo.ca

February 9th, 2026

UNIVERSITY OF WATERLOO

Outline

- Constituency grammar
  - Constituency test
  - Context-free grammars and probabilistic context-free grammars
- A pointer to dependency grammar

Phrase Structures/Constituency Grammar

Constituency grammars focus on the **constituent** relation.

Informally: Sentences have hierarchical structures.

- A sentence is made up of two pieces:
  - Subject, typically a **noun phrase (NP)**
  - Predicate, typically a **verb phrase (VP)**
- NPs and VPs are made up of pieces:
  - a cat = (a + cat)
  - walked to the park = (walk + (to + (the + school)))
  - Each parenthesized phrase is a **constituent** in the **constituent parse.**

What makes a group of words a constituent?

Constituent: a group of words that functions as a single unit.

Linguists (try to) determine constituents via constituency tests.

- A constituency test follows some rules to construct a new sentence, focusing on the constituent candidate of interests.
- If the constructed sentence looks good (to native speakers), we find some evidence about constituency.

Consider this sentence: *Drunks could put off the customers.*

Constituency Test: Coordination

Drunks could put off the customers.

- Coordinate the candidate constituent with something else.
  - Drunks could [put off the customers] and sing.
  - Drunks could put off [the customers] and the neighbors.
  - Drunks [could] and [would] put off the customers.

Constituency Test: Topicalization

Drunks could put off the customers.

- Moving the candidate constituent to the front.
  Modal adverbs can be added to improve naturalness.
  - ... and [the customers], drunks certainly could put off.
  - * ... and [customers], drunks could certainly put off the.

Constituency Test: Deletion

Drunks could put off the customers.

- Delete the span of interest. Word orders can be changed to improve naturalness.
  - Drunks could put off the customers [~~in the bar~~].
  - * Drunks could put off the customers [~~in the~~] bar.

Constituency Test: Substitution

Drunks could put off the customers.

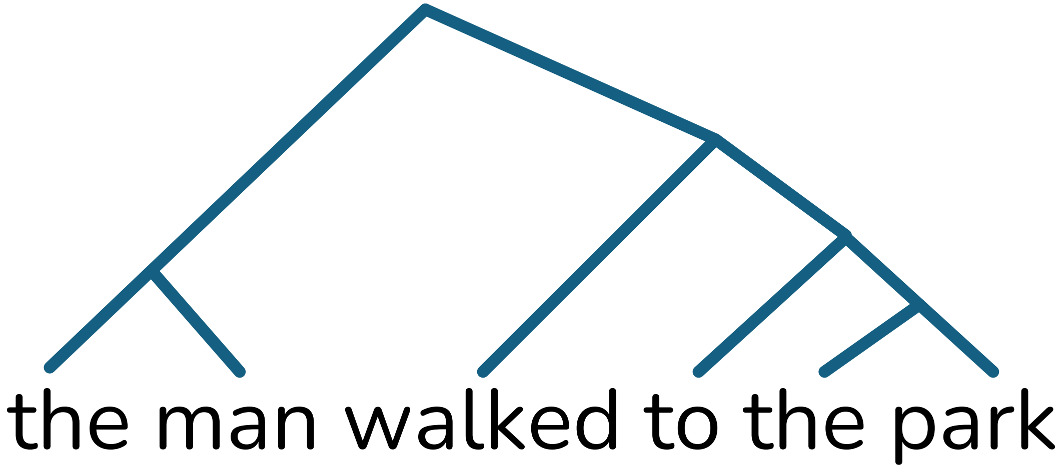
- Substitute the candidate constituent with the appropriate proform (pronoun/proverb/etc.). Slight word order adjustment is allowed to improve naturalness.
  - Drunks could [do so = put off the customers].
  - Drunks could put [them = the customers] off.
  - * Drunks could put the [them = customers] off.

Constituency Parsing as Bracketing

- Brackets: which spans of words are the constituents in a sentence?

- Sentence: *the man walked to the park*
- Bracketing: *((the man) (walked (to (the park))))*

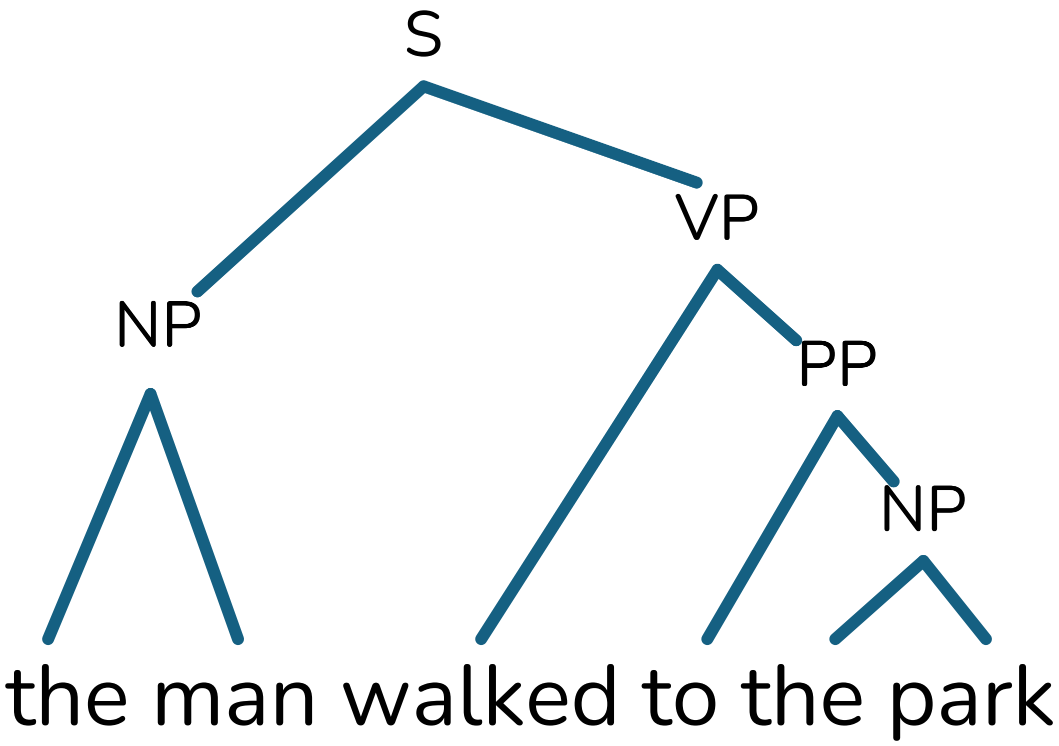
- The brackets can be translated into trees



Labeled Bracketing / Tree

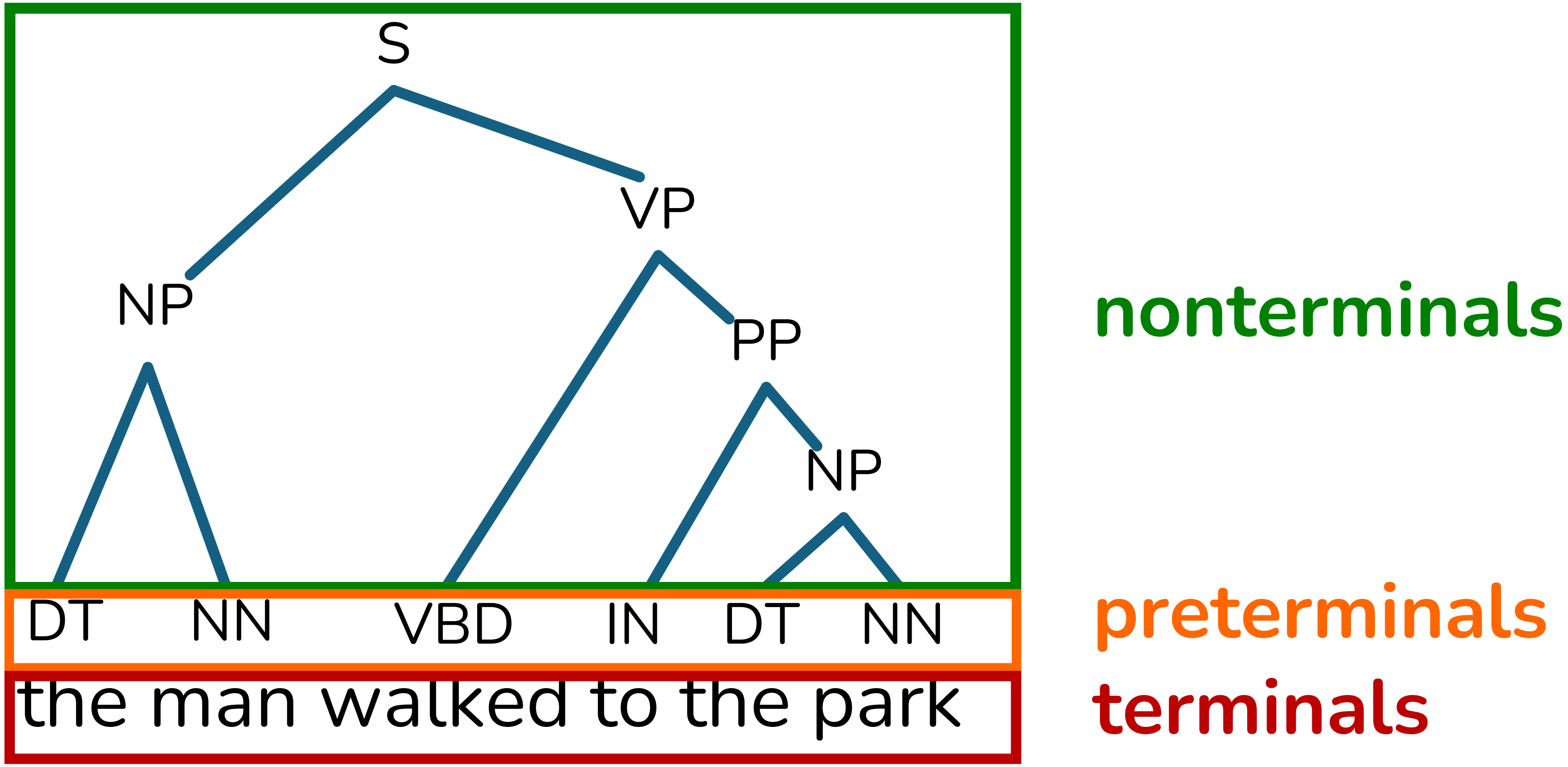
There are categories associated with constituents

(S (NP the man) (VP walked (PP to (NP the park))))



Key:
S = sentence
NP = noun phrase
VP = verb phrase
PP = prepositional phrase

Labeled Bracketing / Tree



Penn Treebank Tagset (Pre-Terminals)

- Marcus et al. (1993): 40K WSJ sentences annotated by linguistic experts.

Tag	Description	Example	Tag	Description	Example
CC	coordin. conjunction	*and, but, or*	SYM	symbol	+,%, &
CD	cardinal number	*one, two*	TO	“to”	*to*
DT	determiner	*a, the*	UH	interjection	*ah, oops*
EX	existential ‘there’	*there*	VB	verb base form	*eat*
FW	foreign word	*mea culpa*	VBD	verb past tense	*ate*
IN	preposition/sub-conj	*of, in, by*	VBG	verb gerund	*eating*
JJ	adjective	*yellow*	VBN	verb past participle	*eaten*
JJR	adj., comparative	*bigger*	VBP	verb non-3sg pres	*eat*
JJS	adj., superlative	*wildest*	VBZ	verb 3sg pres	*eats*
LS	list item marker	*1, 2, One*	WDT	wh-determiner	*which, that*
MD	modal	*can, should*	WP	wh-pronoun	*what, who*
NN	noun, sing. or mass	*llama*	WP$	possessive wh-	*whose*
NNS	noun, plural	*llamas*	WRB	wh-adverb	*how, where*
NNP	proper noun, sing.	*IBM*	$	dollar sign	*$*
NNPS	proper noun, plural	*Carolinas*	#	pound sign	*#*
PDT	predeterminer	*all, both*	“	left quote	‘ or “
POS	possessive ending	*’s*	”	right quote	’ or ”
PRP	personal pronoun	*I, you, he*	(	left parenthesis	[, (, {, <
PRP$	possessive pronoun	*your, one’s*	)	right parenthesis	],), }, >
RB	adverb	*quickly, never*	,	comma	,
RBR	adverb, comparative	*faster*	.	sentence-final punc	. ! ?
RBS	adverb, superlative	*fastest*	:	mid-sentence punc	: ; ... – -
RP	particle	*up, off*			

Penn Treebank Non-Terminal Set

- Marcus et al. (1993): 40K WSJ sentences annotated by linguistic experts.

S	Sentence or clause.
SBAR	Clause introduced by a (possibly empty) subordinating conjunction.
SBARQ	Direct question introduced by a *wh*-word or *wh*-phrase.
SINV	Inverted declarative sentence.
SQ	Inverted yes/no question, or main clause of a *wh*-question.
ADJP	Adjective Phrase.
ADVP	Adverb Phrase.
CONJP	Conjunction Phrase.
FRAG	Fragment.
INTJ	Interjection.
LST	List marker. Includes surrounding punctuation.
NAC	Not A Constituent; used within an NP.
NP	Noun Phrase.
NX	Used within certain complex NPs to mark the head.

PP	Prepositional Phrase.
PRN	Parenthetical.
PRT	Particle.
QP	Quantity Phrase (i.e., complex measure/amount) within NP.
RRC	Reduced Relative Clause.
UCP	Unlike Coordinated Phrase.
VP	Verb Phrase.
WHADJP	*Wh*-adjective Phrase, as in *how hot*.
WHADVP	*Wh*-adverb Phrase.
WHNP	*Wh*-noun Phrase, e.g. *who*, *which book*, *whose daughter*, *none of which*, or *how many leopards*.
WHPP	*Wh*-prepositional Phrase, e.g., *of which* or *by whose authority*.
X	Unknown, uncertain, or unbracketable.

Head of a Constituent

Head of a constituent is the most responsible/important word for the constituent label

Which word makes *the cat* an NP?

- Cat

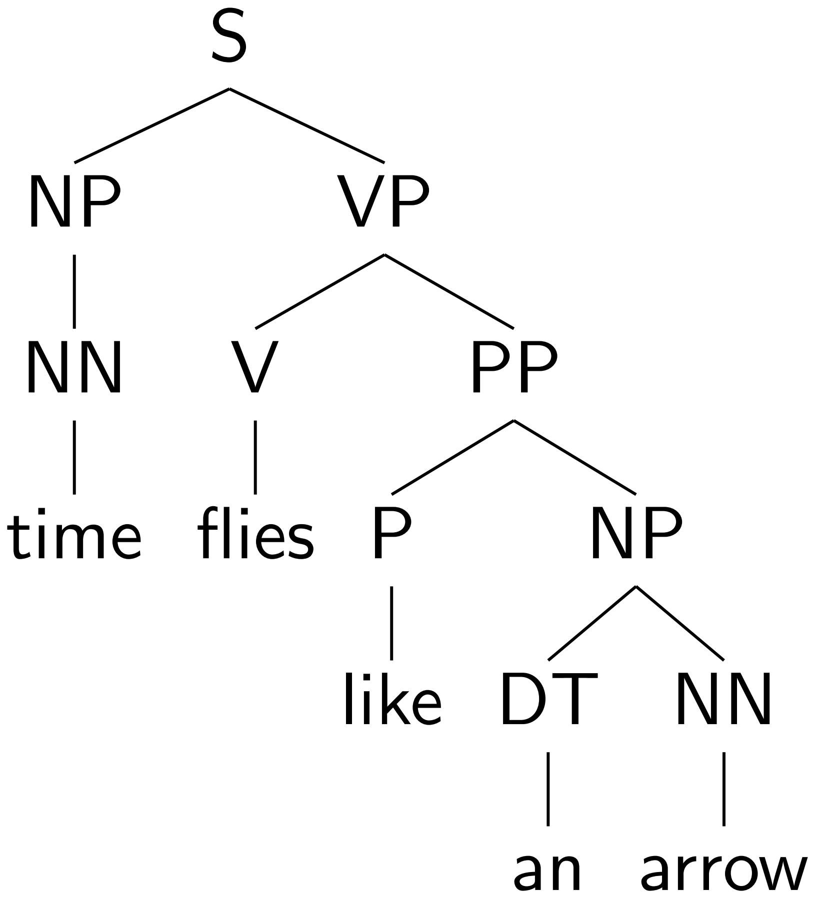
Which word makes *walked to the park* a VP?

- Walked

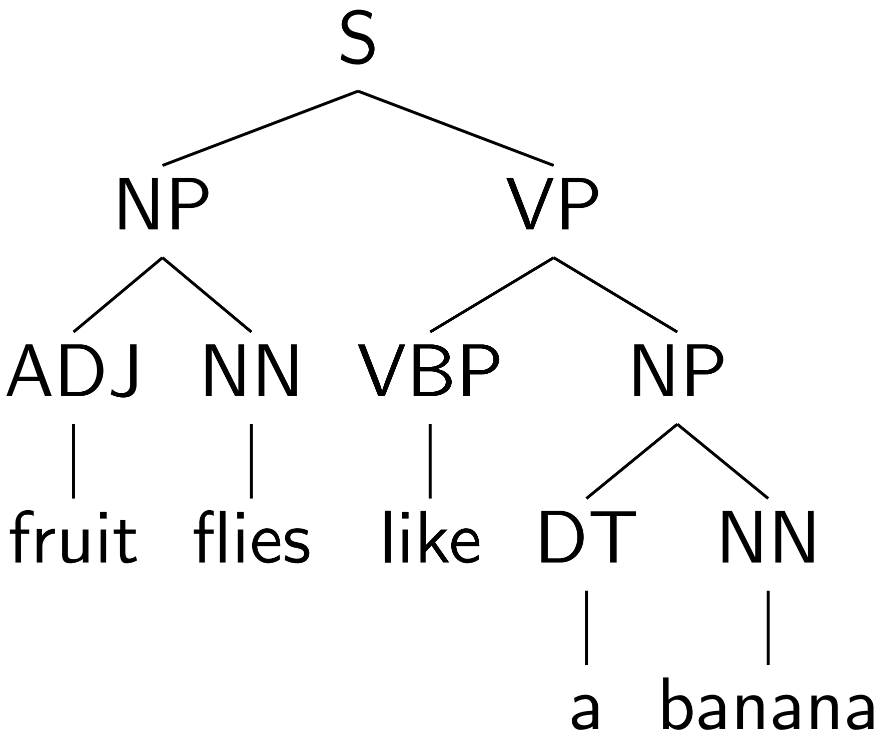
We'll see how this connects to dependency grammar on the last slides.

Syntactic Ambiguities

Time flies like an arrow.



Fruit flies like a banana.



NLP Task: Constituency Parsing

- Given a sentence, output its constituency parse.
- Widely studies task with a rich history.

- Most studies are based on the Penn Treebank.
  (Treebank = corpus of annotated parse trees)

Constituency Parsing within General Formulation

Inference: solve arg max

Modeling: define score function

$$\text{parse}(\mathbf{x}) = \arg\max_{\boldsymbol{y}} \text{score}(\mathbf{x}, \boldsymbol{y}; \mathbf{w})$$

Learning: choose parameter

$\mathbf{x}$: sentence

$\boldsymbol{y}$: constituent parse tree

Constituency Parsing: Modeling

Modeling: define score function

$$\text{parse}(\mathbf{x}) = \arg\max_{\boldsymbol{y}} \text{score}(\mathbf{x}, \boldsymbol{y}; \mathbf{w})$$

- The score of a tree is defined by the sum of constituent scores

$$\text{score}(\mathbf{x}, \boldsymbol{y}; \mathbf{w}) = \sum_{s \in \boldsymbol{y}} \text{span-score}(s; \mathbf{x}, \mathbf{w})$$

Constituency Parsing: Modeling and Learning

$$\text{score}(\mathbf{x}, \boldsymbol{y}; \mathbf{w}) = \sum_{s \in \boldsymbol{y}} \text{span-score}(s; \mathbf{x}, \mathbf{w})$$

Each span score can be modeled with a neural network.

Training objective (Kitaev and Klein, 2018): let the collection of the true spans have the highest accumulated span scores among all parses.

$$\text{Loss } \mathcal{L}(\mathbf{w}) = \max\left(0, \boxed{\sum_{s \in \hat{\boldsymbol{y}}} \text{span-score}(s; \mathbf{x}, \mathbf{w})} - \sum_{s \in \boldsymbol{y}} \text{span-score}(s; \mathbf{x}, \mathbf{w}) + \delta\right)$$

$\hat{\boldsymbol{y}}$: Tree with the highest score

How do we calculate this?

Constituency Parsing: Inference

Inference: solve $\arg\max$

$$\text{parse}(\mathbf{x}) = \arg\max_{\boldsymbol{y}} \text{score}(\mathbf{x}, \boldsymbol{y}; \mathbf{w})$$

- Let's first assume $\boldsymbol{y}$ is a binary unlabeled parse tree.
  (We'll see any tree can be converted into binary without loss of geneality)
  - Each node is either a terminal node or the parent of two other nodes.
  - There is one root node.

Constituency Parsing: Inference

$$\text{score}(\mathbf{x}, \boldsymbol{y}; \mathbf{w}) = \sum_{s \in \boldsymbol{y}} \text{span-score}(s; \mathbf{x}, \mathbf{w})$$

- Each node is either a terminal node or the parent of two other nodes
- There must be a constituent whose children are two terminal nodes.

$f_{\ell,r}$: The maximum sum of subtree scores if $[\ell, r]$ is a constituent.

The (Simplified) CKY Algorithm

$$\text{score}(\mathbf{x}, \boldsymbol{y}; \mathbf{w}) = \sum_{s \in \boldsymbol{y}} \text{span-score}(s; \mathbf{x}, \mathbf{w})$$

$f_{\ell,r}$: The maximum sum of subtree scores if $[\ell, r]$ is a constituent.

$f_{1,n}$: The maximum possible sum of subtree scores if the sentence is fully parsed.

$$f_{\ell,r} = \text{score}_{l,r} + \max_{l \leq m < r} \{f_{l,m} + f_{m+1,r}\}$$

Edge case: $f_{i,i} = 0$.

Introduced by Cocke, Younger and Kasami independently in 1960-1970s.

Context-Free Grammar (CFG)

A generative way to describe constituency parsing.

- A CFG defines some "rewrite rules" to rewrite nonterminals as other nonterminals or terminals

S → NP VP

"S goes to NP VP"

NP → DT NN

VP → VBD PP

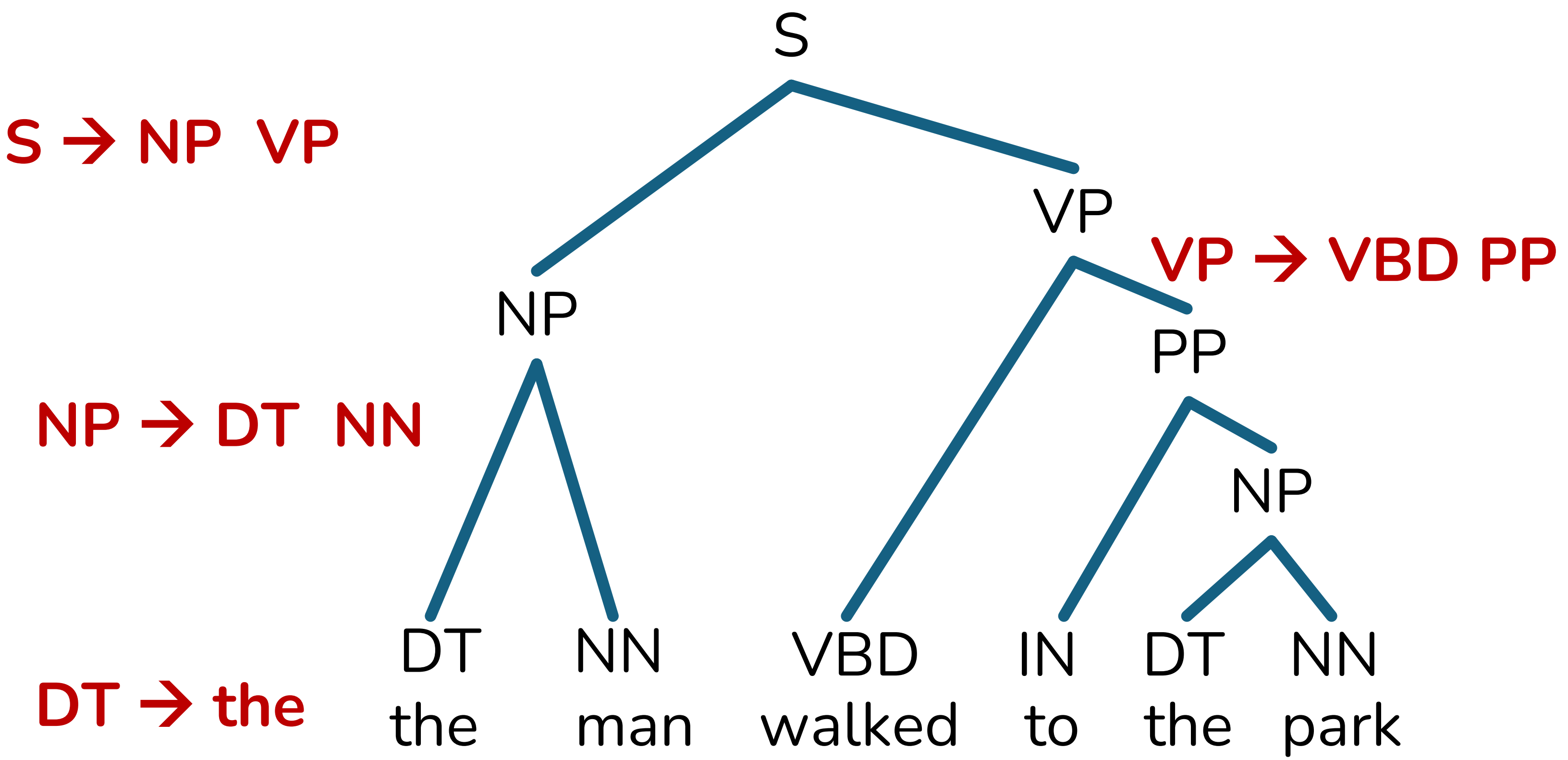
PP → IN NP

NN → man

DT → the

Context-Free Grammar (CFG)

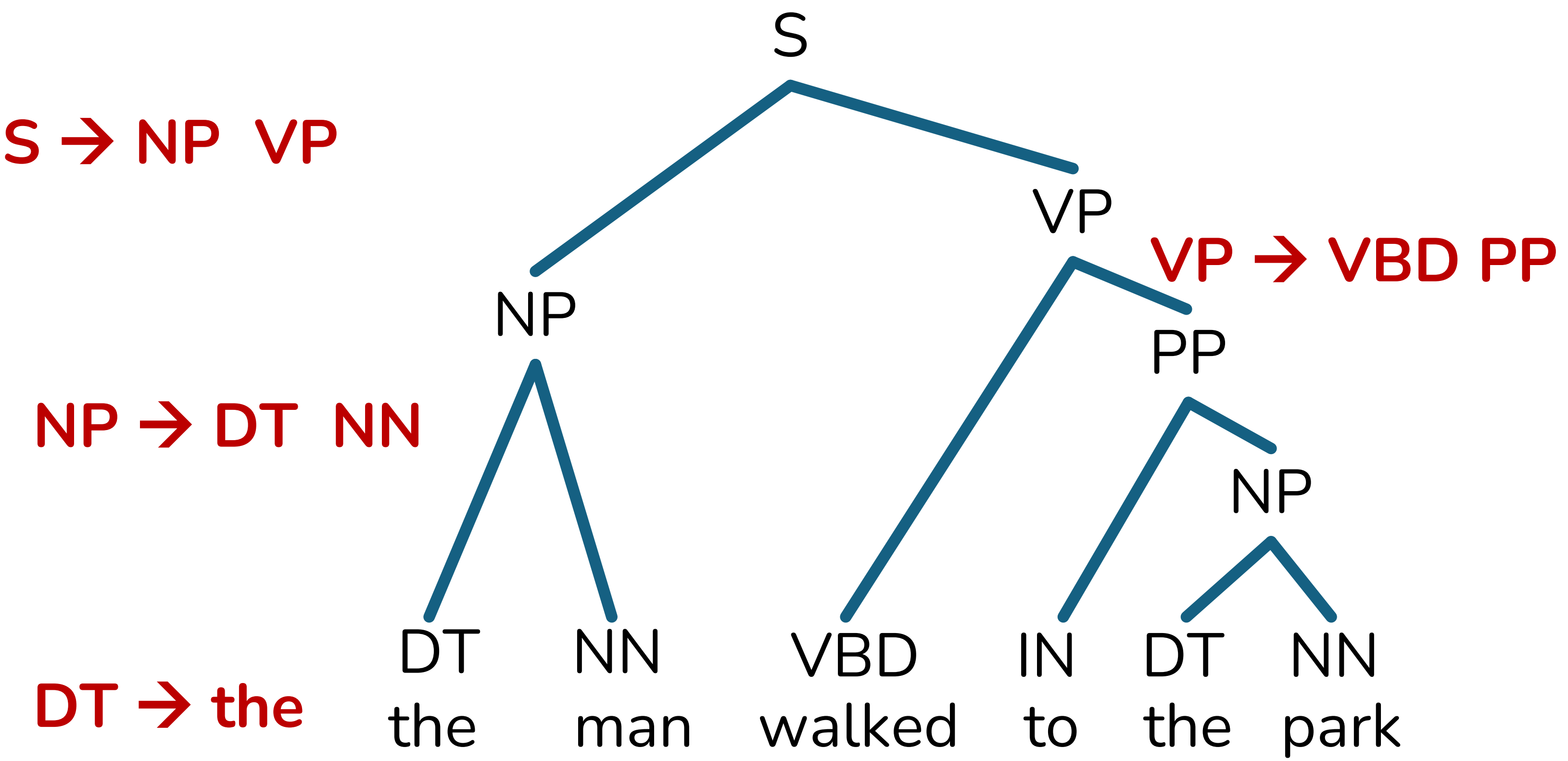
A sequence of rewrites corresponds to a bracketing (induces a hierarchical tree structure).



Why Context-Free?

A rule to rewrite NP does not depend on the context of that NP.

The left-hand side (LHS) of a rule is only a single non-terminal (without any other context).



Probabilistic Context-Free Grammar (PCFG)

- Assign probabilities to rewrite rules:

NP → DT NN	0.5
NP → NNS	0.3
NP → NP PP	0.2

same nonterminal can be on both left and right sides

- Probabilities must sum to 1 for each left-hand side nonterminal.
- Given a sentence s and its tree T, the probability of generating s with rules T in grammar $\mathcal{G}$ is $P(s,T;\mathcal{G}) = \prod_{r \in T} p(r)$, where r denotes a rule.

PCFG: Learning

Given a treebank, what is the MAP estimation of the PCFG?

$$p(A \to B) = \frac{\text{count}(A \to B)}{\text{count}(A)}$$

$$p(A \to B\ C) = \frac{\text{count}(A \to B\ C)}{\text{count}(A)}$$

- A PCFG assigns probabilities to
  - Sequences of rewrite operations that terminate in terminals---this sequence implies the natural-language “yield”.
  - Bracketings of sentences.

CKY with PCFG Formalism

- Find the max-probability tree for a sentence

$$\text{score}(\mathbf{x}, \boldsymbol{y}; \mathbf{w}) = \log \prod_{r \in \boldsymbol{y}} p(r; w_r) \qquad \text{yield}(\boldsymbol{y}) = \mathbf{x}$$

$$= \sum_{r \in \boldsymbol{y}} \log p(r; w_r)$$

↑ rules applied in generating the parse tree

CKY with PCFG Formalism

$f_{\ell,r,b}$: the maximum possible log probability that words within range $[\ell, r]$ are the outcome of a nonterminal label b.

$$f_{\ell,r,b} = \max_{(b \to a \ \ c) \in \mathcal{R}_b} \max_{\ell \leq m < r} \log p(b \to a \ \ c) + f_{\ell,m,a} + f_{m+1,r,c}$$

Rules to rewrite b

midpoint for splitting

Edge case: set the appropriate $f_{i,i,\ell} = 0$ when word i could have label ℓ, and otherwise $-\infty$.

See SLP: Chapter 18 for a running example.

Inside Algorithm

- Find the probability for generating a sequence from a certain non-terminal (counting all possible trees).

$$P(s_{\ell..r}, b) = \sum_{(b \to a\ c) \in \mathcal{R}_b} \sum_{\ell \leq m < r} p(b \to a\ c) P(s_{\ell..m}, a) P(s_{m+1..r}, c)$$

Rules to rewrite b

midpoint for splitting

The Chomsky Normal Form (CNF)

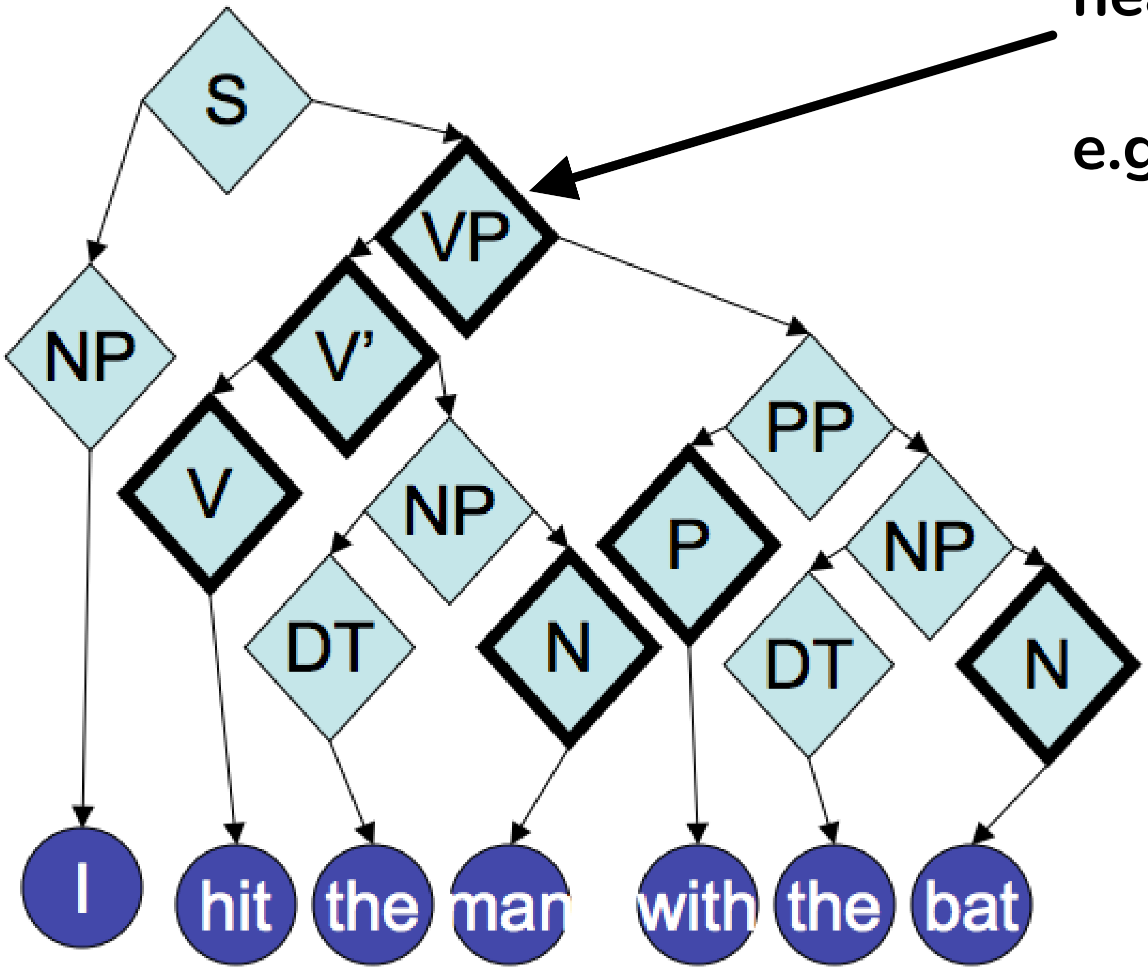
- For any free-form PCFG, there exists an equivalent PCFG in which each nonterminal has zero or two nonterminal children, or it directly goes to a terminal.
- Trees satisfying the latter conditions are said to be in the Chomsky normal form.

A → B C D	0.5
A → B C	0.5

added dummy node ↙

A → B C-D	0.5
A → B C	0.5
C-D → C D	1.0

From Constituency to Dependency

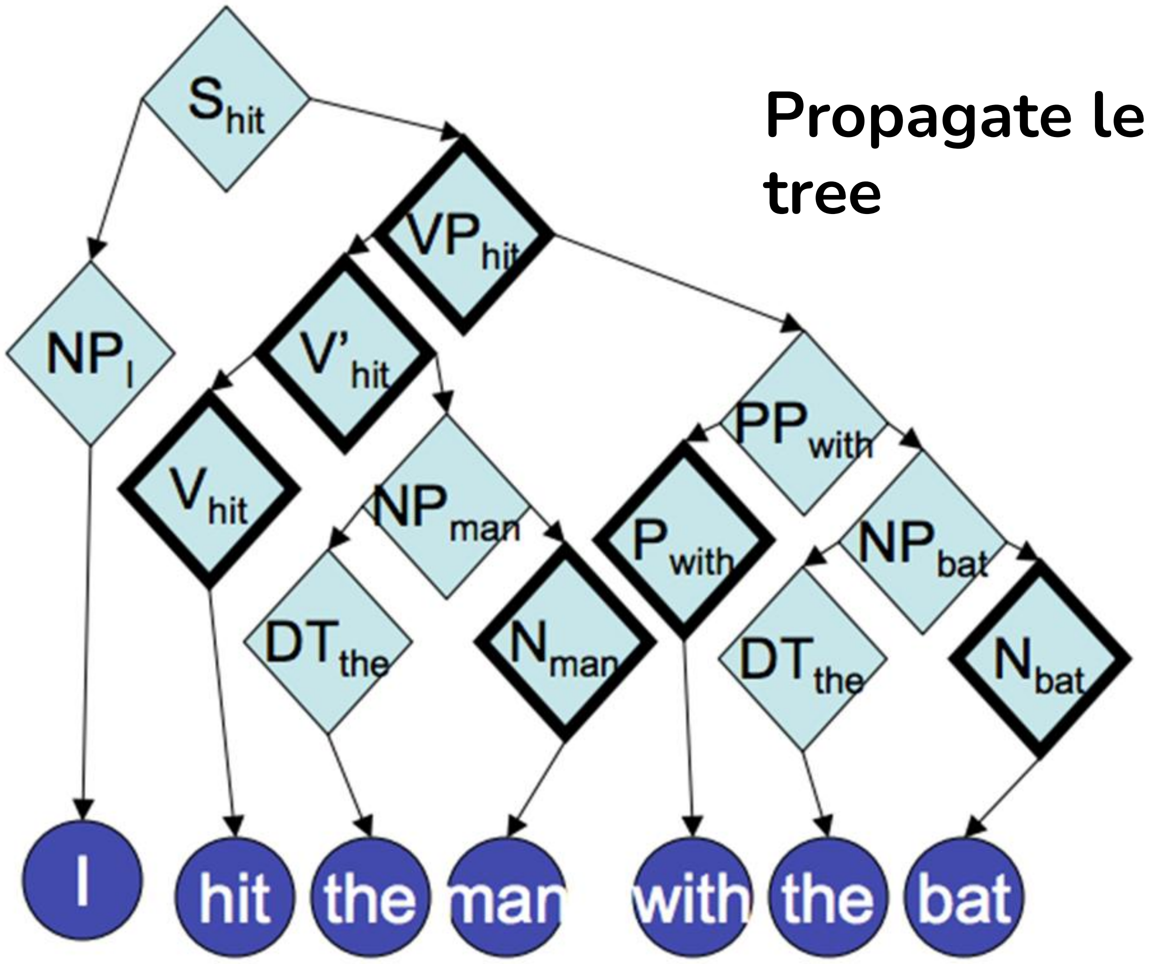


heads have bold outline

e.g., VP is head of S -> NP VP

from Noah Smith

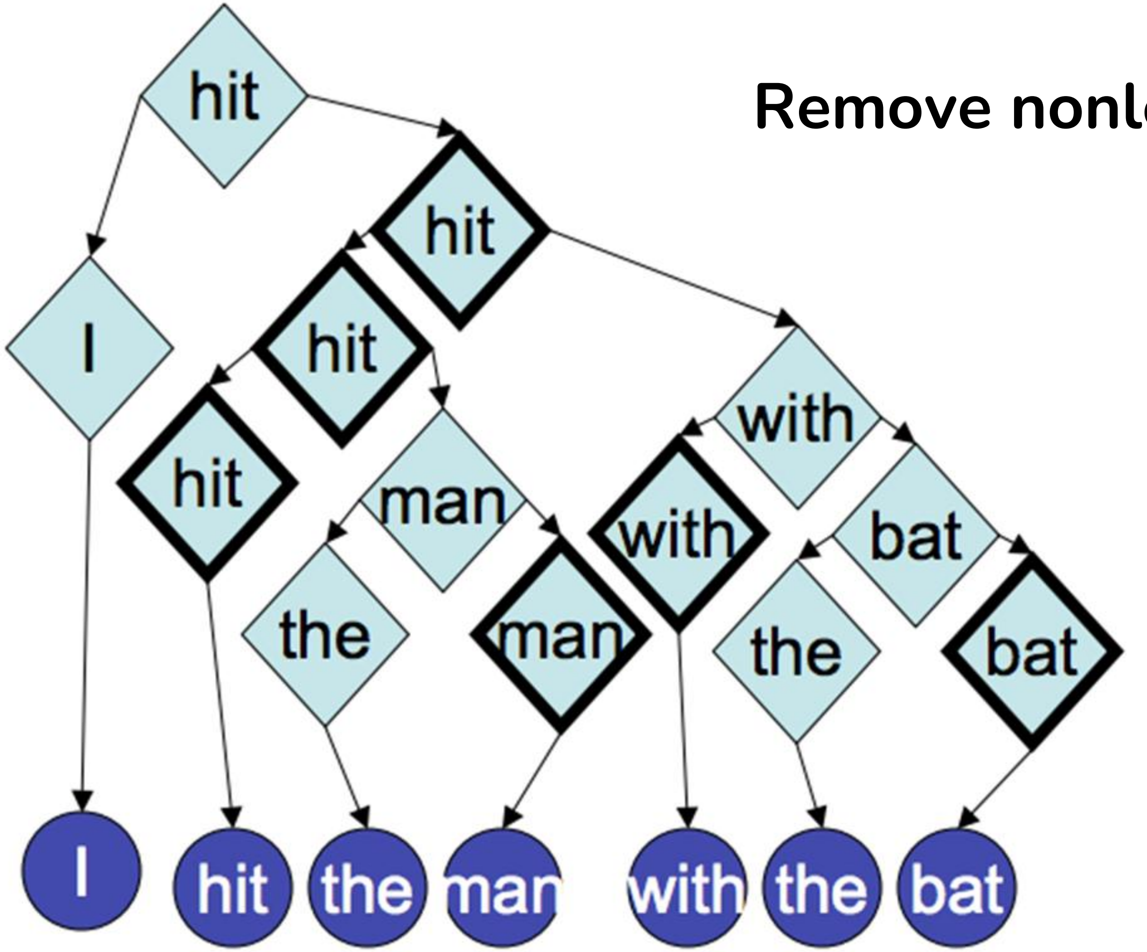
From Constituency to Dependency



Propagate lexical heads up the tree

from Noah Smith

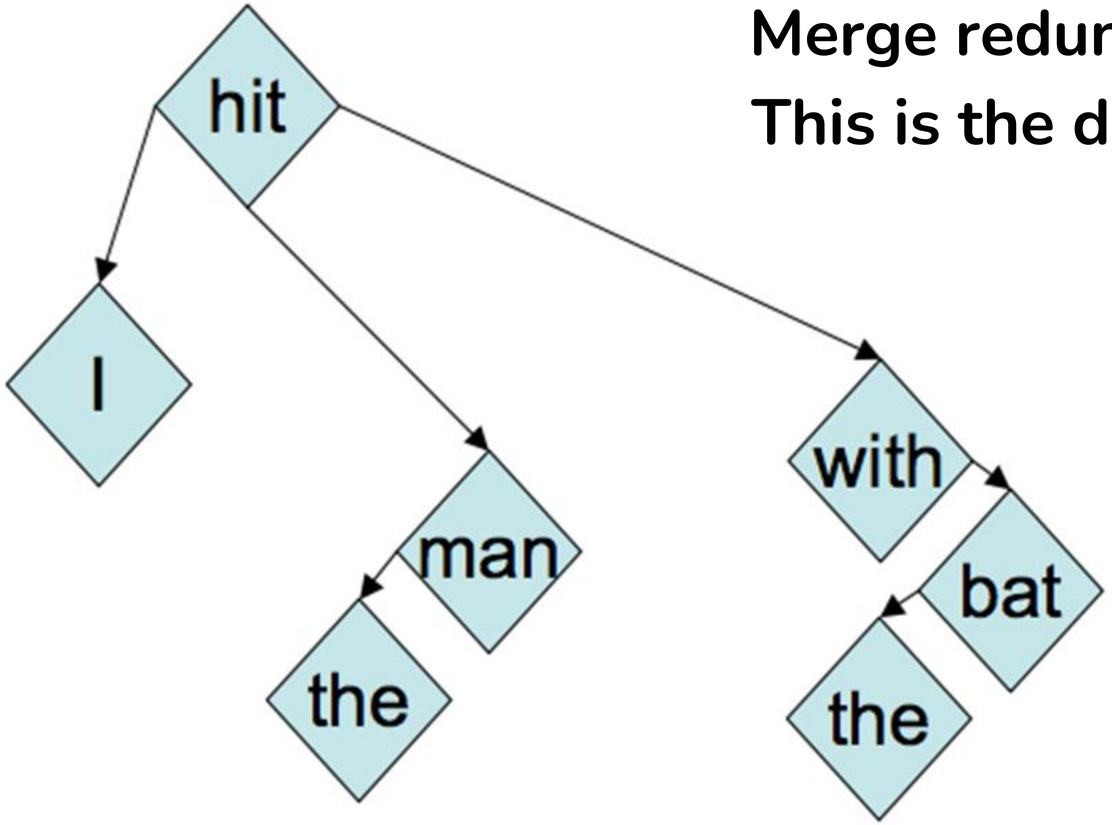
From Constituency to Dependency



Remove nonlexical parts

from Noah Smith

From Constituency to Dependency



Merge redundant nodes

This is the dependency parse tree

from Noah Smith

Dependency Parses

- Directly model the relation between words.
- See SLP: Chapter 19 for more details.

Next

- Multimodal language models