

(Advanced) Natural Language Processing

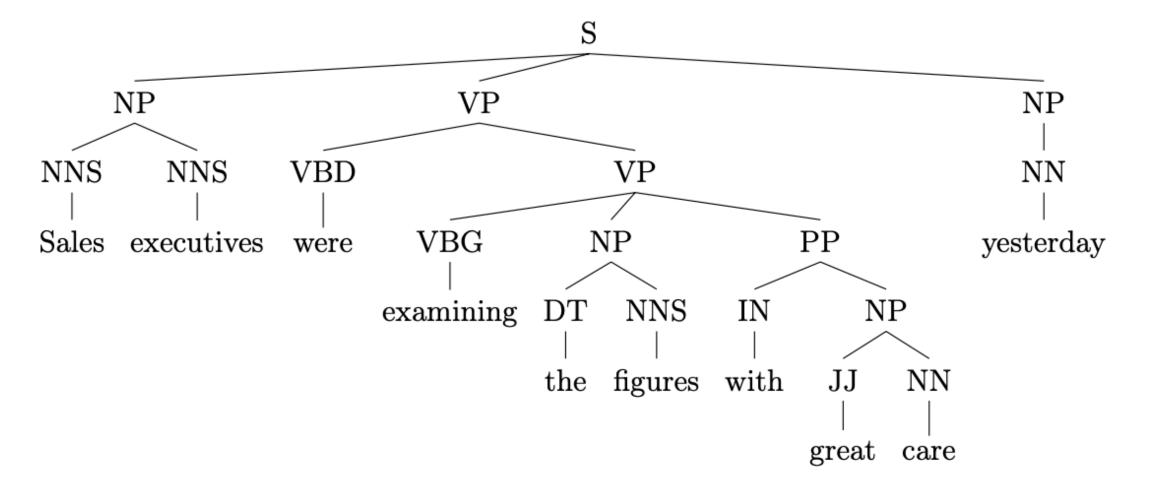
# LI4: Dependency Parsing

### COS 484/584

Spring 2021

## Constituency parsing (cont'd)

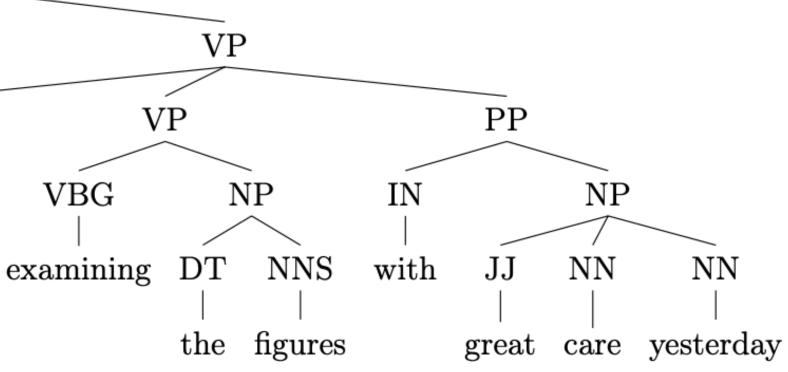
Gold: (1, 10, S), (1, 2, NP), (3, 9, VP), (4, 9, VP), (5, 6, NP), (7, 9, PP), (8, 9, NP), (10, 10, NP)



Predicted: (1, 10, S), (1, 2, NP), (3, 10, VP), (4, 6, VP), (5, 6, NP), (7, 10, PP), (8, 10, NP)  $\mathbf{S}$ NPNNS VBD NNS

**a** 1

Sales executives were



## Evaluating constituency parsing

- Recall: (# correct constituents in candidate) / (# constituents in gold tree)
- Precision: (# correct constituents in candidate) / (# constituents in candidate)
- Labeled precision/recall require getting the non-terminal label correct
- F1 is the harmonic mean of precision and recall = (2 \* precision \* recall) / (precision + recall)• Part-of-speech tagging accuracy is evaluated separately

## Zoom poll

Gold: (1, 10, S), (1, 2, NP), (3, 9, VP), (4, 9, VP), (5, 6, NP), (7, 9, PP), (8, 9, NP), (10, 10, NP) Predicted: (1, 10, S), (1, 2, NP), (3, 10, VP), (4, 6, VP), (5, 6, NP), (7, 10, PP), (8, 10, NP)

What are the **labeled** precision (P) / recall (R) in the above example?

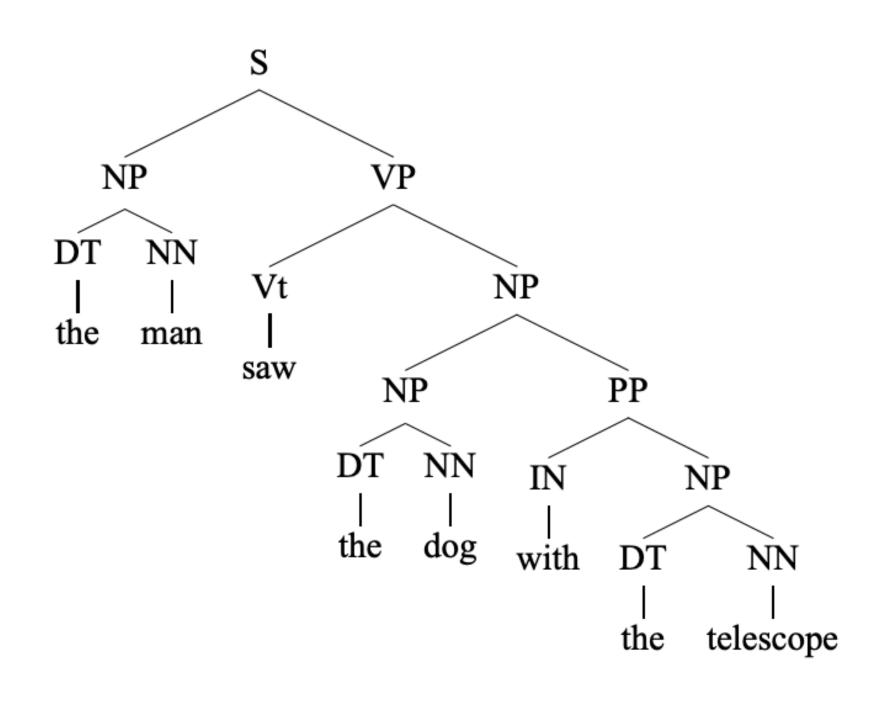
(a) P = 3/8, R = 3/7(b) P = 3/7, R = 3/8(c) P = 1/2, R = 1/2(d) P = 1, R = 1

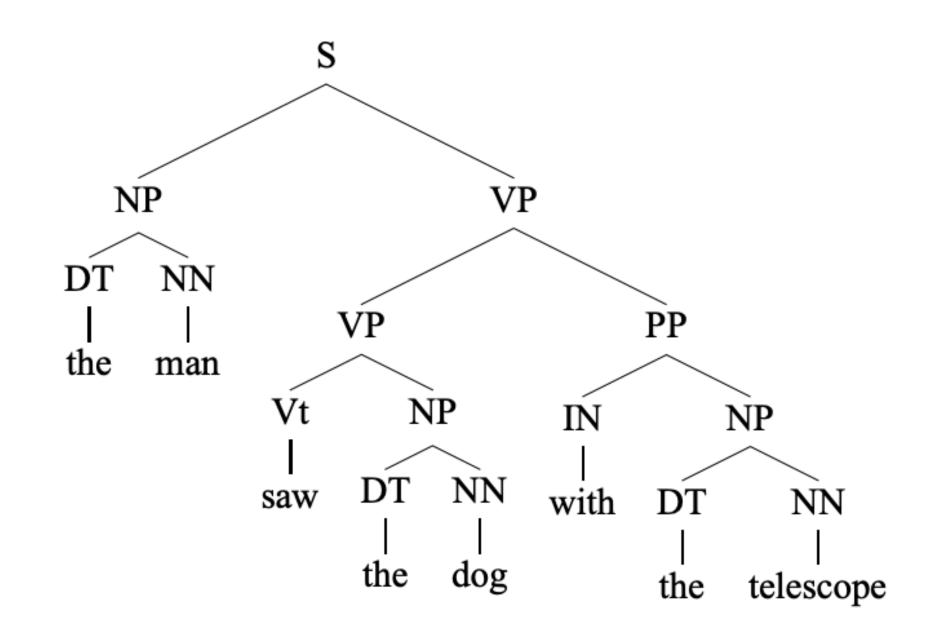
The answer is (b). F1 = 40%, tagging accuracy = 100%



## Weaknesses of PCFGs

Lack of sensitivity to lexical information (words)





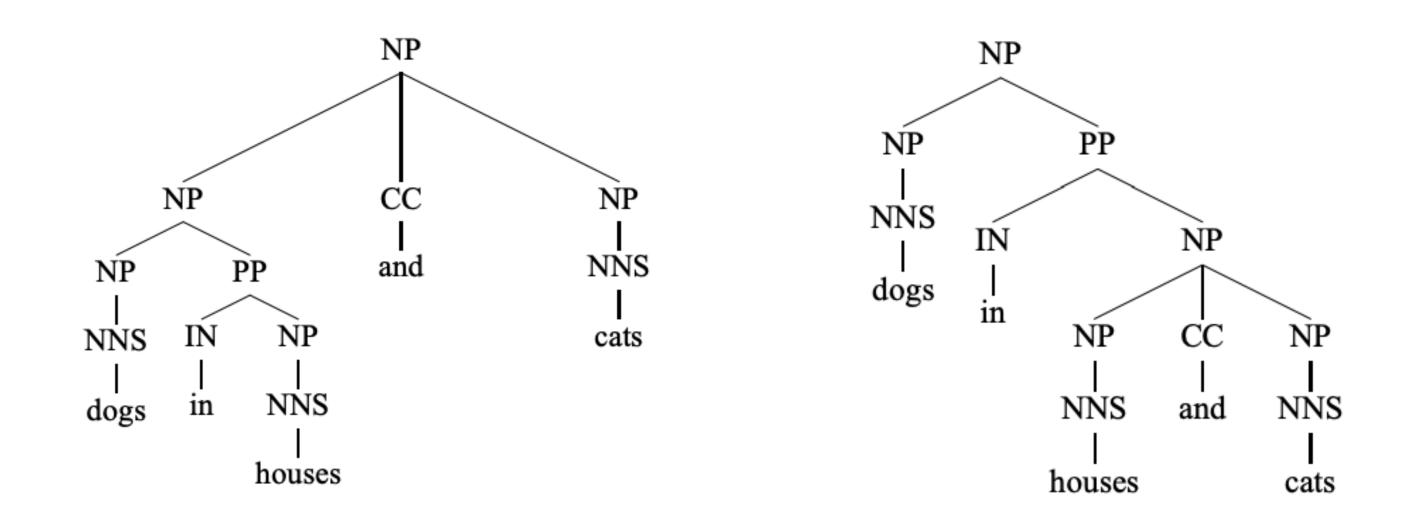
The only difference between these two parses:

 $q(\text{NP} \rightarrow \text{NP} \text{PP}) \text{ vs } q(\text{VP} \rightarrow \text{VP} \text{PP})$ 

#### ... without looking at the words!

## Weaknesses of PCFGs

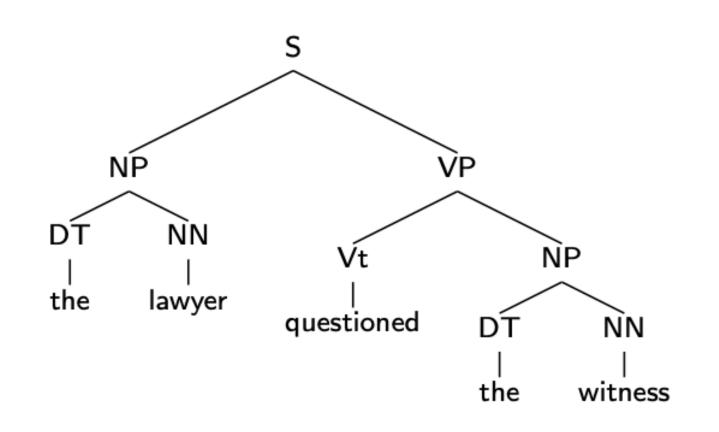
Lack of sensitivity to lexical information (words)



Exactly the same set of context-free rules!

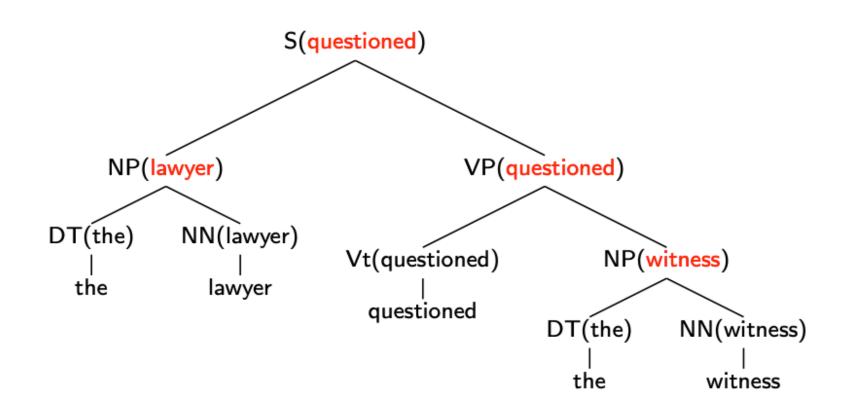
## Lexicalized PCFGs [advanced]

### • Key idea: add **headwords** to trees



• Each context-free rule has one special child that is the head of the rule (a core idea in syntax)

$$\begin{array}{ccccccccc} S & \Rightarrow & NP & VF \\ VP & \Rightarrow & Vt & NF \\ NP & \Rightarrow & DT & NF \end{array}$$



(VP is the head) D (Vt is the head) D (NN is the head) N NN

The headwords are decided by manual rules!



## Lexicalized PCFGs [advanced]

S(saw)	$\rightarrow_2$	NP(man)	ν
VP(saw)	$\rightarrow_1$	Vt(saw)	Ν
NP(man)	$\rightarrow_2$	DT(the)	Ν
NP(dog)	$\rightarrow_2$	DT(the)	Ν
Vt(saw)	$\rightarrow$	saw	
DT(the)	$\rightarrow$	the	
NN(man)	$\rightarrow$	man	
NN(dog)	$\rightarrow$	dog	

- Results for a PCFG: 70.6% recall, 74.8% precision
- Results for a lexicalized PCFG: 88.1% recall, 88.3% precision

VP(saw) NP(dog) NN(man) NN(dog)

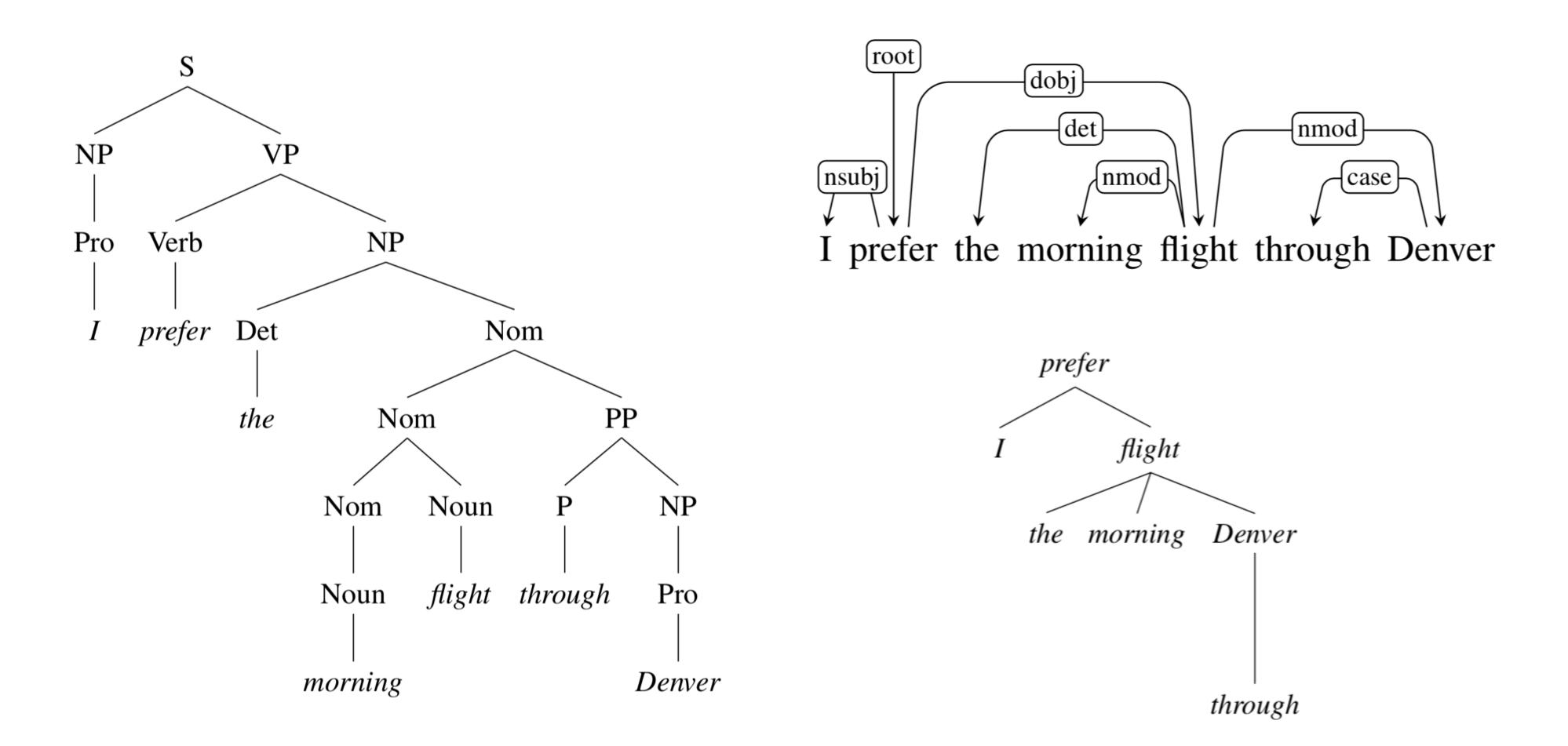
## Constituency vs dependency parsing

- Constituency structure
- Context-free grammar (CFG)
- Probabilistic context-free grammar (PCFG)
- Treebanks
- The CKY algorithm
- Evaluation
- Lexicalized PCFGs

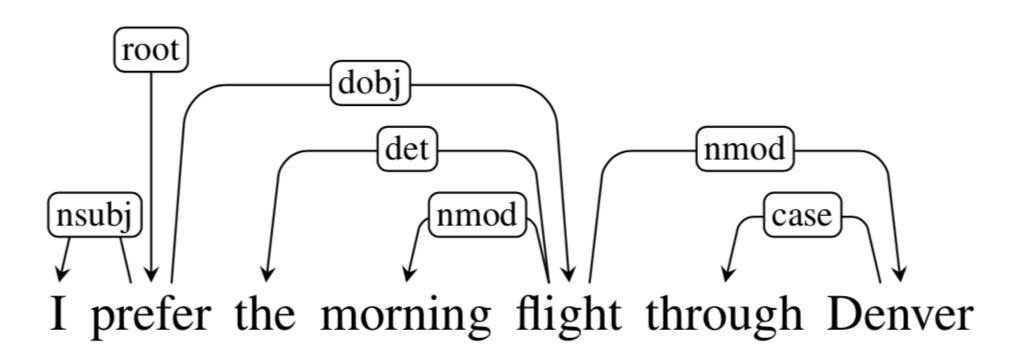


- Dependency structure
- The Arc-standard algorithm
- Dependency treebanks
- Evaluation

### Constituency vs dependency structure



## Dependency structure



- called **dependencies**
- object, apposition, etc)
- The arrow connects a **head** (governor) and a **dependent** (modifier)
- Usually, dependencies form a tree

Consists of relations between lexical items, normally *binary*, *asymmetric* relations ("arrows")

• The arrows are commonly **typed** with the name of grammatical relations (subject, prepositional

## Dependency relations

<b>Clausal Argument Relations</b>	Description	
NSUBJ	Nominal subject	
DOBJ	Direct object	
IOBJ	Indirect object	
CCOMP	Clausal complement	
XCOMP	Open clausal complement	
<b>Nominal Modifier Relations</b>	Description	
NMOD	Nominal modifier	
AMOD	Adjectival modifier	
NUMMOD	Numeric modifier	
APPOS	Appositional modifier	
DET	Determiner	
CASE	Prepositions, postpositions and other case markers	
Other Notable Relations	Description	
CONJ	Conjunct	
CC	Coordinating conjunction	
Figure 14.2 Selected dependency relations from the Universal Dependency set. (de Marn-		

effe et al., 2014)

## Dependency relations

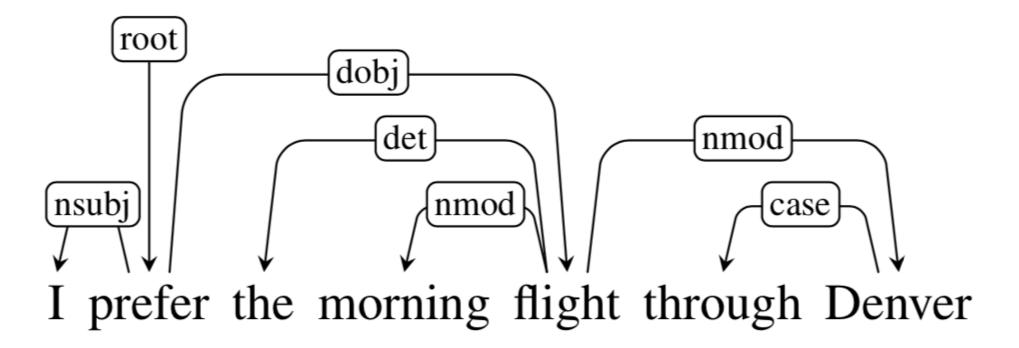
Relation	Examples with
NSUBJ	United cancele
DOBJ	United diverted
	We booked her
IOBJ	We booked her
NMOD	We took the <b>m</b>
AMOD	Book the cheap
NUMMOD	Before the stor
APPOS	<i>United</i> , a <b>unit</b> (
DET	<b>The</b> <i>flight</i> was
	Which <i>flight</i> w
CONJ	We flew to Den
CC	We flew to Der
CASE	Book the flight
Figure 14.3	Examples of core Unive
	-

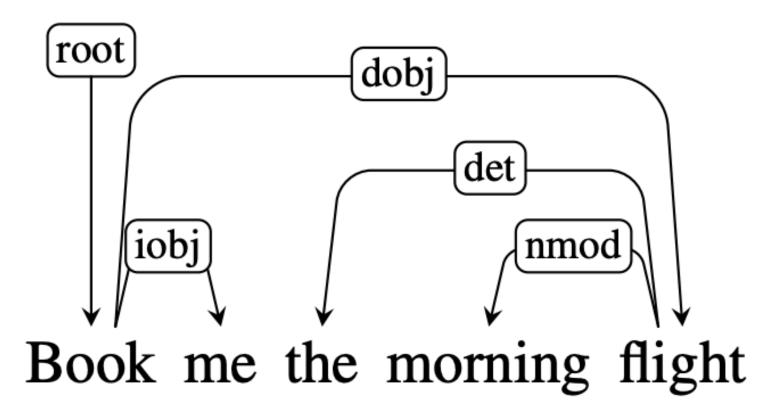
- head and dependent ed the flight. d the flight to Reno. the first **flight** to Miami. **r** the flight to Miami. orning flight. **pest** *flight*. rm JetBlue canceled **1000** *flights*. of UAL, matched the fares. canceled. vas delayed? nver and **drove** to Steamboat. nver and *drove* to Steamboat. t through Houston.
- versal Dependency relations.

## Dependency structure: more examples

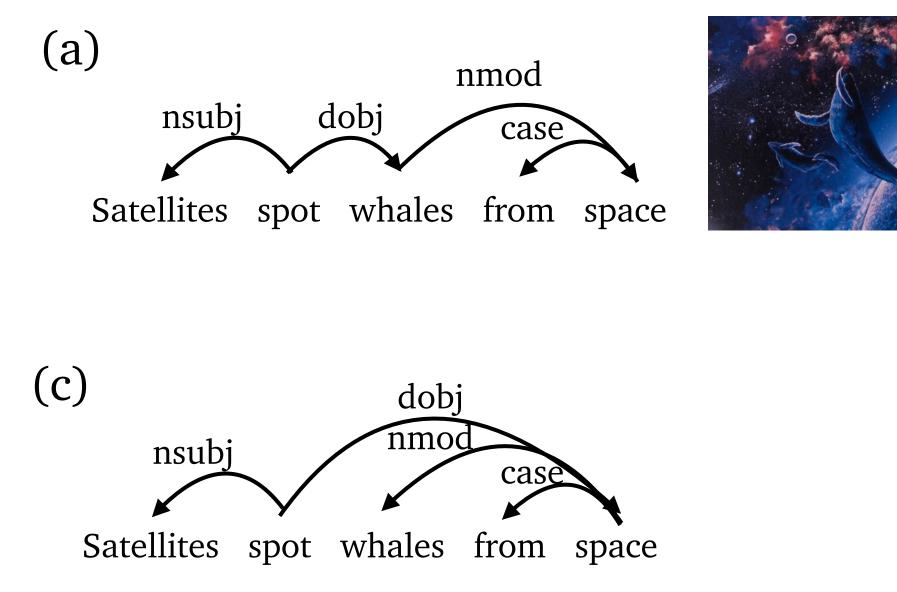
#### I prefer the morning flight through Denver

Book me the morning flight





# Zoom poll

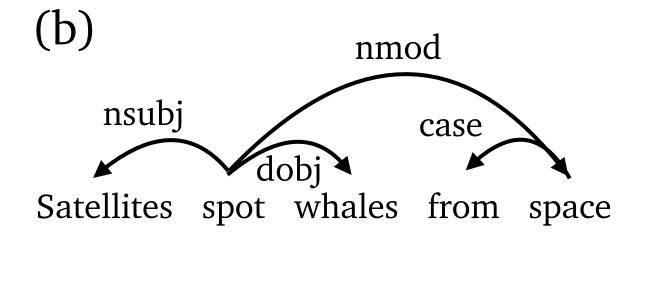


The answer is (b).

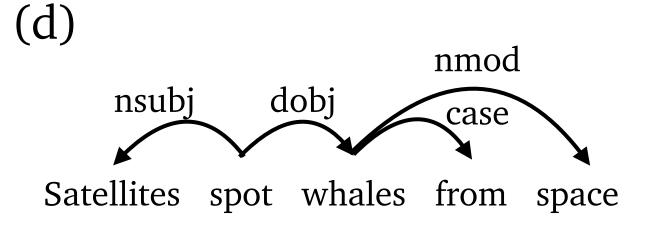


#### Which of the following is the correct dependency structure for "Satellites spot whales from space"?











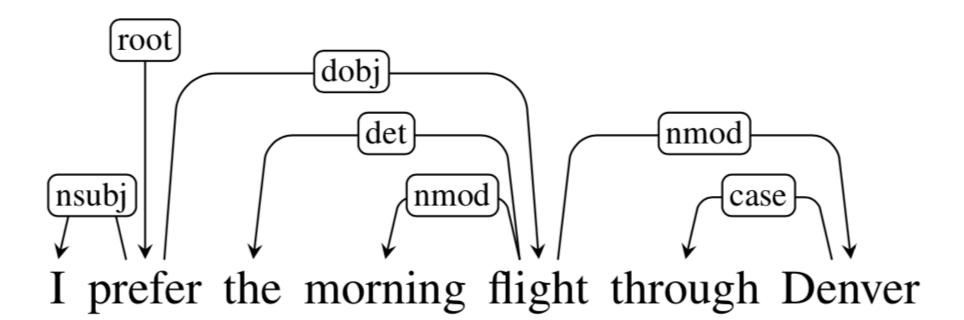
## Dependency parsing

Syntactic parsing is the task of recognizing a sentence and assigning a structure to it. **Dependency** parsing is the task of recognizing a sentence and assigning a **dependency** structure to it.

Input

I prefer the morning flight through Denver

### Output

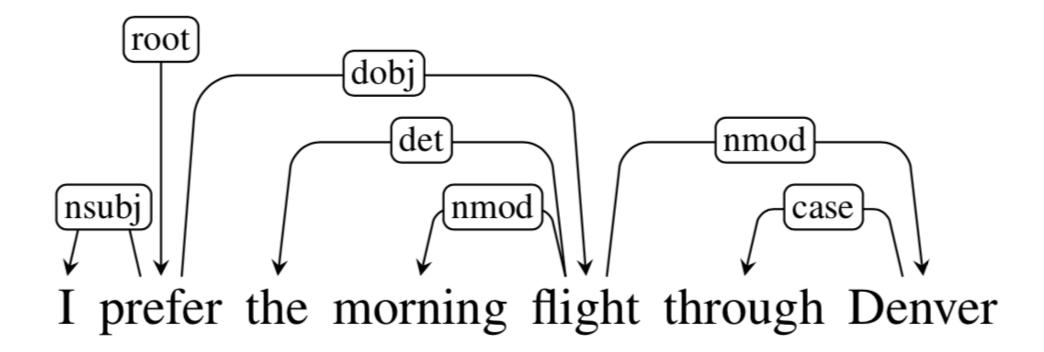


## Dependency formalisms

Usually a tree structure

- There is only one root
- Every word except for the root has one head (parent)
- No cycles:  $A \longrightarrow B$ ,  $B \longrightarrow C$ ,  $C \longrightarrow A$

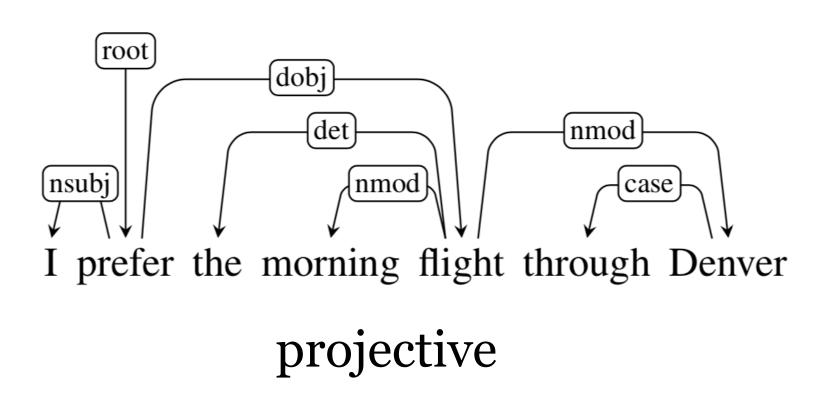
• Alternatively, we can just add a fake node ROOT, so each word has exactly one head



## Dependency formalisms

### Additional constraint: **projectivity**

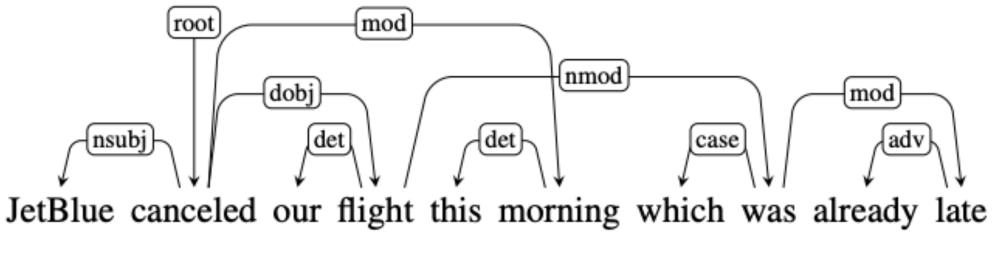
linear order, with all arcs above the words



Non-projectivity arises due to long distance dependencies or in languages with flexible word order.

We only focus on projective parsing

#### • **Definition**: there are no crossing dependency arcs when the words are laid out in their



#### non-projective

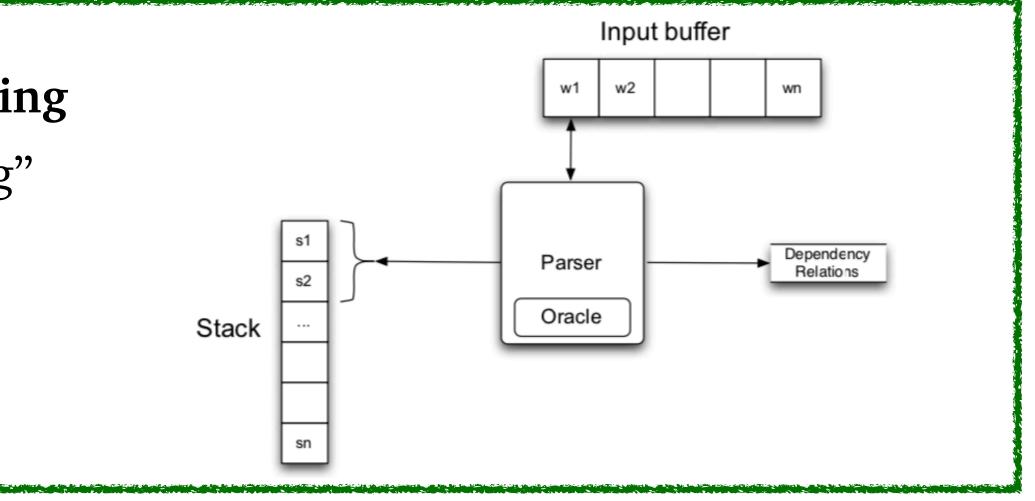
Dataset	# Sentences	(%) Projective
English	39,832	99.9
Chinese	16,091	100.0
Czech	72,319	76.9
German	38,845	72.2

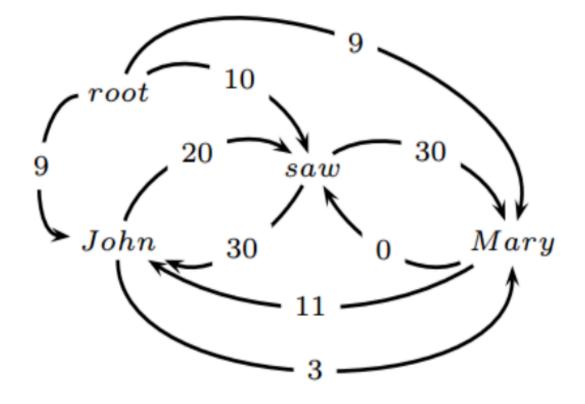
## Two families of algorithms

#### **Transition-based dependency parsing**

• Also called "shift-reduce parsing"

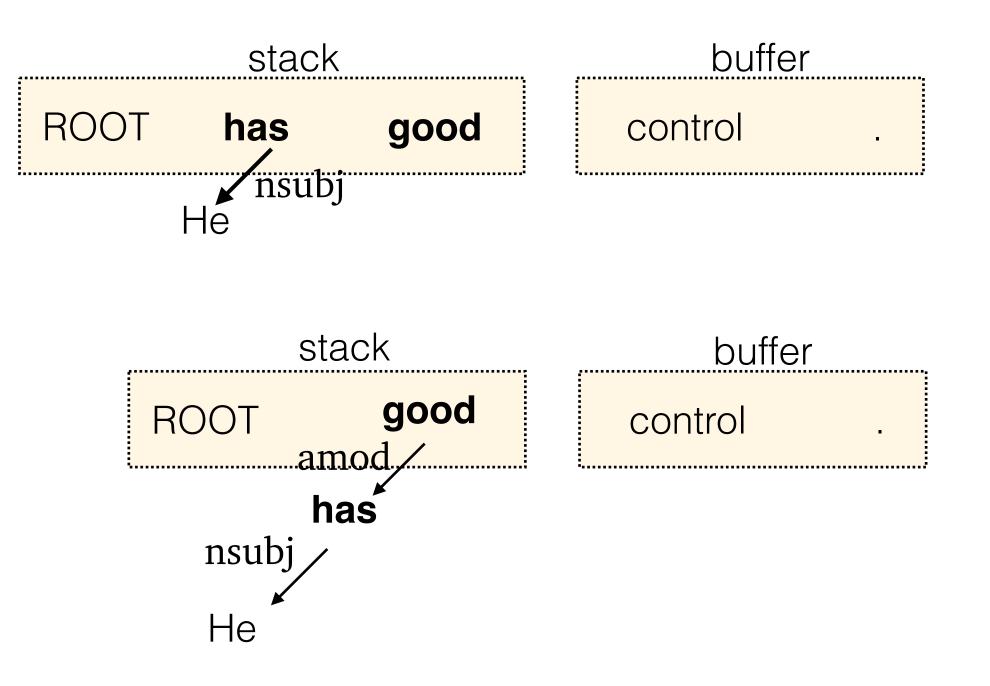
### Graph-based dependency parsing





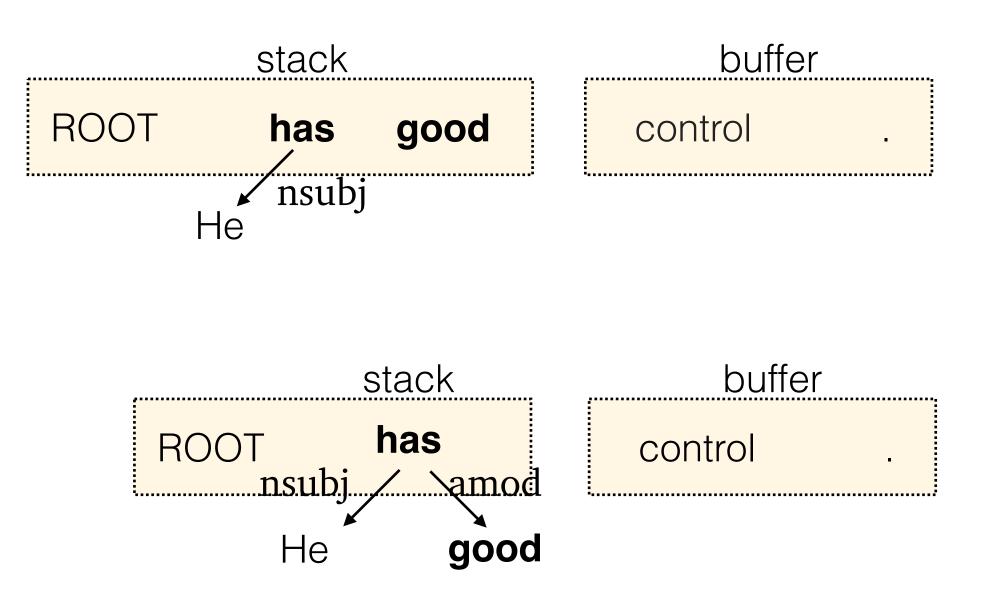
- Given: a sentence of  $w_1, w_2, ..., w_n$
- The parsing process is modeled as a sequence of transitions
- A configuration consists of a stack s, a buffer b and a set of dependency arcs A:
  c = (s, b, A)
- Initially,  $s = [ROOT], b = [w_1, w_2, ..., w_n], A = \emptyset$
- Three types of transitions: LEFT-ARC (*r*), RIGHT-ARC (*r*), SHIFT I will define them in the next slides!
- A configuration is terminal if s = [ROOT] and  $b = \emptyset$

 $s_1, s_2$ : the top 2 words on the stack ( $s_1 = \text{good}, s_2 = \text{has}$ );  $b_1$ : the first word in the buffer ( $b_1$  = control)

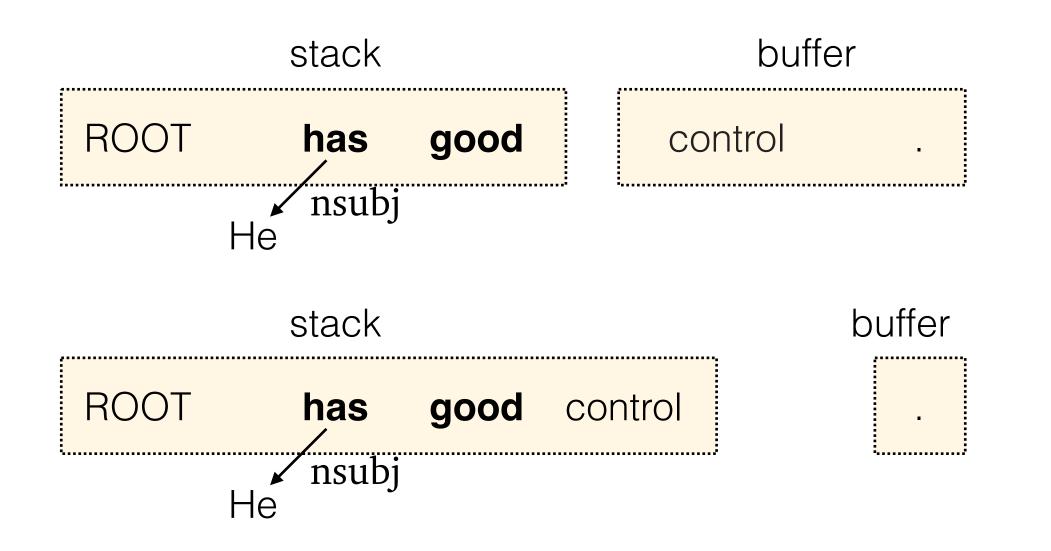


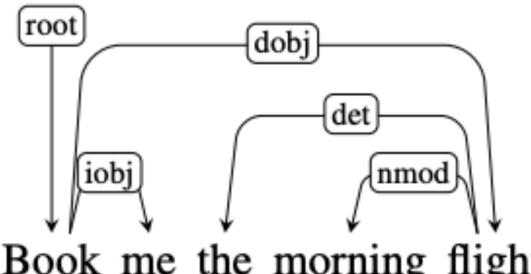
LEFT-ARC (*r*): add an arc ( $s_1 \xrightarrow{r} s_2$ ) to *A*, remove  $s_2$  from the stack

 $s_1, s_2$ : the top 2 words on the stack ( $s_1 = \text{good}, s_2 = \text{has}$ );  $b_1$ : the first word in the buffer ( $b_1$  = control) **RIGHT-ARC** (*r*): add an arc ( $s_2 \xrightarrow{r} s_1$ ) to *A*, remove  $s_1$  from the stack



 $s_1, s_2$ : the top 2 words on the stack ( $s_1 = \text{good}, s_2 = \text{has}$ );  $b_1$ : the first word in the buffer ( $b_1$  = control) **SHIFT**: move  $b_1$  from the buffer to the stack



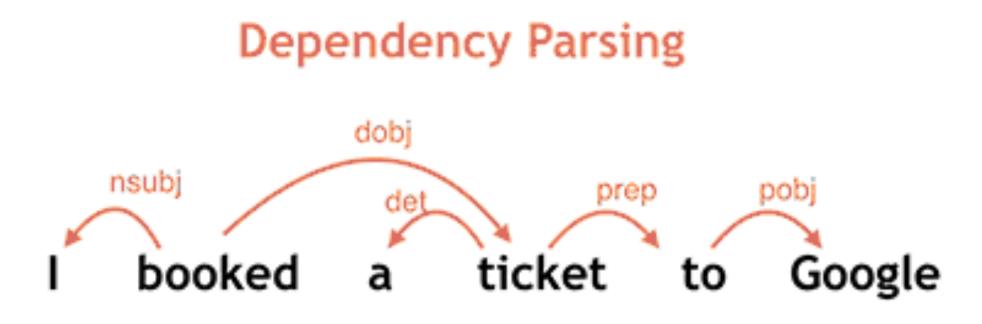


### "Book me the morning flight" A running example

Book me the morning flight

	stack	buffer	action	added arc
0	[ROOT]	[Book, me, the, morning, flight]	SHIFT	
1		[me, the, morning, flight]	SHIFT	
2	[ROOT, Book, me]	[the, morning, flight]	RIGHT-ARC(iobj)	(Book, iobj, me)
3		[the, morning, flight]	SHIFT	
4	[ROOT, Book, the]	[morning, flight]	SHIFT	
5	[ROOT, Book, the, morning]	[flight]	SHIFT	
6	[ROOT, Book, the,morning,flight]		LEFT-ARC(nmod)	(flight,nmod,morning)
7	[ROOT, Book, the, flight]	[]	LEFT-ARC(det)	(flight,det,the)
8	[ROOT, Book, flight]	[]	RIGHT-ARC(dobj)	
9	[ROOT, Book]	[]	RIGHT-ARC(root)	
10	[ROOT]	[]		

## Transition-based dependency parsing



https://ai.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html

## Zoom poll

Which of the following transition sequences is correct for the sentence "He likes dogs"?

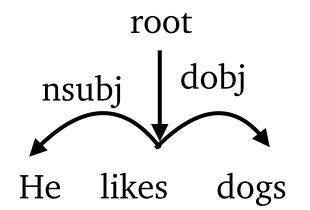
(a) SHIFT, SHIFT, RIGHT-ARC(dobj), SHIFT, LEFT-ARC(nsubj), RIGHT-ARC(root)

(b) SHIFT, SHIFT, SHIFT, RIGHT-ARC(dobj), LEFT-ARC(nsubj), RIGHT-ARC(root)

(c) SHIFT, SHIFT, LEFT-ARC(nsubj), SHIFT, RIGHT-ARC(dobj), RIGHT-ARC(root)

(d) SHIFT, SHIFT, SHIFT, LEFT-ARC(nsubj), RIGHT-ARC(dobj), RIGHT-ARC(root)

Both (b) and (c) are correct.





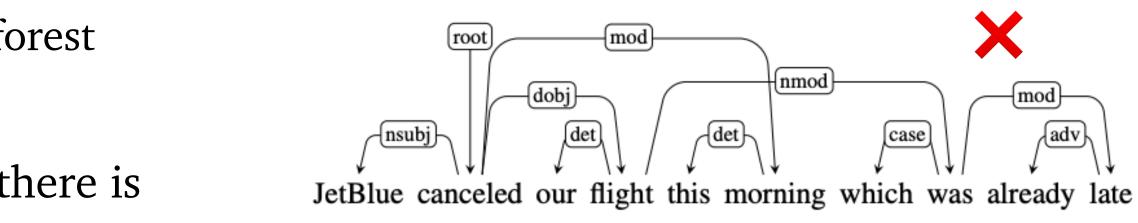
### Transition-based dependency parsing

Given: a sentence of  $w_1, w_2, \ldots, w_n$ 

Q: How many transitions are needed? How many times of SHIFT?

#### **Correctness** [advanced]

- For every complete transition sequence, the resulting graph is a projective dependency forest (soundness)
- For every projective dependency tree G, there is a transition sequence that generates G (completeness)



However, one parse tree can have multiple valid transition sequences.

### How to decide which transitions to take?

Key idea: we can learn a statistical machine learning model from dependency treebanks!

- The major English dependency treebank: converting from Penn Treebank using rule-based algorithms
  - $\bullet$

### Universal Dependencies

Universal Dependencies (UD) is a framework for consistent annotation of grammar (parts of speech, morphological features, and syntactic dependencies) across different human languages. UD is an open community effort with over 300 contributors producing nearly 200 treebanks in over 100 languages. If you're new to UD, you should start by reading the first part of the Short Introduction and then browsing the annotation guidelines.



(De Marneffe et al, 2006): Generating typed dependency parses from phrase structure parses (Johansson and Nugues, 2007): Extended Constituent-to-dependency Conversion for English

• Universal Dependencies: nearly 200 treebanks in 100 languages were collected since 2016

https://universaldependencies.org/

#### Current UD Languages

Information about language families (and genera for families with multiple branches) is mostly taken from WALS Online (IE = Indo-European).

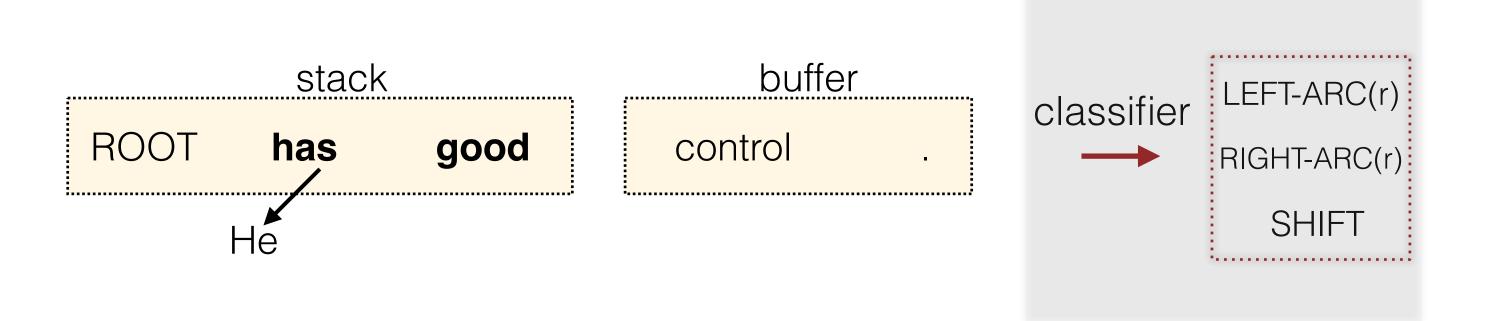
Afrikaans    1    49K    40    IE, Germanic      Akkadian    2    23K    10    Afro-Asiatic, Semitic      Akkadian    1    <1K    100    Tupian, Tupari      Albanian    1    <1K    100    Abanian      Amharic    1    10K    4700    Afro-Asiatic, Semitic      Anniaric    1    10K    4700    Afro-Asiatic, Semitic      Ancient Greek    2    416K    400    IE, Greek      Ancient Greek    2    416K    400    Arawakan      Arabic    3    1,042K    100    Arawakan      Araprina    1    <1K    100    Arawakan      Arserian    1    13K    100    Afro-Asiatic, Semitic      Arawakan    1    13K    100    Afro-Asiatic, Semitic      Bambara    1    13K    100    Basque      Basque    1    121K    100    Basque      Basque    1    10K    4700,5%    IE, Indic      Bujgarian    1    10K    4700,5%    IE, Slavic		100	Abaza	1	<1K	P	Northwest Caucasian
Akkadian    2    23K    23K    23K    Afro-Asiatic, Semitic      Akuntsu    1    <1K	$\rightarrow$	_		1			
Akuntsu    1    <1K	$\rightarrow$		Akkadian	2	23K		
Albanian    1    <1K			Akuntsu	1	<1K	<b>8</b> 0	Tupian, Tupari
Ancient Greek    2    416K    400    IE, Greek      Apurina    1    <1K	$\rightarrow$	_	Albanian	1	<1K	W	IE, Albanian
Image: Solution of the second seco	-		Amharic	1	10K		Afro-Asiatic, Semitic
Image: Second	-	12	Ancient Greek	2	416K	<b>4</b> 20	IE, Greek
Armenian    1    52K    52K <t< td=""><td>-</td><td><b>6</b></td><td>Apurina</td><td>1</td><td>&lt;1K</td><td><b>E0</b></td><td>Arawakan</td></t<>	-	<b>6</b>	Apurina	1	<1K	<b>E0</b>	Arawakan
Non-Assyrian    1    <1K	$\rightarrow$	ø	Arabic	3	1,042K	eiW	Afro-Asiatic, Semitic
Bambara    1    13K    13K    Mande      Basque    1    121K    Basque    Basque      Belarusian    1    275K    4<10,5    IE, Slavic      Bojpuri    2    6K    100    IE, Indic      Belarusian    1    10K    4/10,5    IE, Celtic      Belarusian    1    10K    4/10,5    Mongolic      Berton    1    10K    4/10,5    Mongolic      Bulgarian    1    156K    4/10,6    Mongolic      Buryat    1    10K    4/10,7    Mongolic      Catalan    1    531K    100    IE, Romance      Chukchi    1    6K    9    Chukotko-Kamchatkan      Chukchi    1    6K    9    Chukotko-Kamchatkan      Sioo-Tibetan    233K    6    Sino-Tibetan      Sioo-Tibetan    233K    6    Sino-Tibetan      Sioo-Tibetan    1    19K    20W    IE, Slavic      Coptic    1    48K    46    Afro-Asiatic, Egyptian      Czech    2,227K	$\rightarrow$		Armenian	1	52K	# <b>#</b> %<@6	IE, Armenian
Bambara    1    13K    13K    Mande      Basque    1    121K    Basque    Basque      Belarusian    1    275K    Amorian State    Basque      Belarusian    1    275K    Amorian State    IE, Slavic      Belarusian    1    275K    Amorian State    IE, Slavic      Belarusian    1    275K    Amorian State    IE, Slavic      Belarusian    1    10K    Amorian State    IE, Celtic      Bulgarian    1    156K    Amorian State    IE, Slavic      Buryat    1    10K    Amorian State    Mongolic      Image: Cantonese    1    13K    O    Sino-Tibetan      Image: Catalan    1    531K    Image: Catalan    IE, Romance      Image: Chukchi    1    6K    O    Sino-Tibetan      Image: Classical Chinese    1    233K    O    Sino-Tibetan      Image: Classical Chinese    1    233K    Image: Classical Chinese    1    233K      Image: Classical Chinese    1    233K    Image: Classie Classical Chinese </td <td><math>\rightarrow</math></td> <td>X</td> <td>Assyrian</td> <td>1</td> <td>&lt;1K</td> <td><b>E0</b></td> <td>Afro-Asiatic, Semitic</td>	$\rightarrow$	X	Assyrian	1	<1K	<b>E0</b>	Afro-Asiatic, Semitic
Belarusian1275KImage: Constraint of the second sec			Bambara	1	13K	<b>111</b>	Mande
Bhojpuri    2    6K    IE, Indic      Breton    1    10K    IE, Celtic      Bulgarian    1    156K    IE, Slavic      Buryat    1    10K    IE, Slavic      Cantonese    1    13K    Mongolic      Catalan    1    531K    IE, Romance      Chinese    5    285K    IE, Octive Sino-Tibetan      Chinese    5    285K    Image Sino-Tibetan      Chukchi    1    6K    Chukotko-Kamchatkan      Classical Chinese    1    233K    Sino-Tibetan      Coptic    1    48K    Image Sino-Tibetan      Danish    2    2,227K    Image Sino-Tibetan      Dutch    2    306K    Image Sino-Tibetan			Basque	1	121K	DI	Basque
Breton    1    10K    #**@0.fW    IE, Celtic      Bulgarian    1    156K    #*@    IE, Slavic      Buryat    1    10K    #*@    Mongolic      Cantonese    1    13K    O    Sino-Tibetan      Catalan    1    531K    IE, Romance    Sino-Tibetan      Catalan    1    531K    IE, Romance    Sino-Tibetan      Catalan    1    6K    O    Chukotko-Kamchatkan      Chinese    5    285K    Image W    Sino-Tibetan      Classical Chinese    1    233K    O    Sino-Tibetan      Coptic    1    48K    Image W    IE, Slavic      Coptic    1    48K    Image W    IE, Slavic      Image W    IE, Slavic    IE, Slavic    Image W      Image W    IE, Germanic    Image M    Image M    Image			Belarusian	1	275K		IE, Slavic
Bulgarian    1    156K    Image: Constant of the set o		•	Bhojpuri	2	6K	<b>80</b>	IE, Indic
Buryat    1    10K    Image: Constant of the set of th		200	Breton	1	10K	<b>e</b> %egstw	IE, Celtic
Image: Cartonese    1    13K    Image: Sino-Tibetan      Image: Cartalan    1    531K    Image: Cartalan    IE, Romance      Image: Chinese    5    285K    Image: Cartalan    IE, Romance      Image: Chinese    5    285K    Image: Cartalan    IE, Romance      Image: Chinese    5    285K    Image: Cartalan    Image: Chinese      Image: Chinese    1    6K    Chukotko-Kamchatkan      Image: Classical Chinese    1    233K    Image: Chinese    Sino-Tibetan      Image: Classical Chinese    1    233K    Image: Chinese    Sino-Tibetan      Image: Classical Chinese    1    233K    Image: Chinese    Sino-Tibetan      Image: Classical Chinese    1    48K    Image: Chinese    Sino-Tibetan      Image: Coptic    1    48K    Image: Chinese    Afro-Asiatic, Egyptian      Image: Croatian    1    199K    Image: Chinese    IE, Slavic      Image: Croatian    1    199K    Image: Chinese    IE, Germanic      Image: Croatian    2    100K    Image: Chinese    IE, Germanic			Bulgarian	1	156K		IE, Slavic
Catalan    1    531K    IE, Romance      Chinese    5    285K    Impow    Sino-Tibetan      Chukchi    1    6K    Chukotko-Kamchatkan      Classical Chinese    1    233K    Could for the set of the s	-	*	Buryat	1	10K		Mongolic
Chinese    5    285K    Imp W    Sino-Tibetan      Chukchi    1    6K    Chukotko-Kamchatkan      Classical Chinese    1    233K    Coptic    Sino-Tibetan      Coptic    1    48K    Sino-Tibetan    Afro-Asiatic, Egyptian      Coptic    1    199K    Sino-W    IE, Slavic      Croatian    1    199K    CopticW    IE, Slavic      Czech    5    2,227K    CopticW    IE, Slavic      Danish    2    100K    Coptic    IE, Germanic      Dutch    2    306K    Imp W    IE, Germanic	-	*	Cantonese	1	13K	P	Sino-Tibetan
Chukchi    1    6K    Chukotko-Kamchatkan      Classical Chinese    1    233K    Sino-Tibetan      Coptic    1    48K    Afro-Asiatic, Egyptian      Croatian    1    199K    IE, Slavic      Czech    5    2,227K    Cietter      Danish    2    100K    IE, Germanic      Dutch    2    306K    IE, Germanic			Catalan	1	531K	DI	IE, Romance
Image: Sine Classical Chinese    1    233K    Image: Sine Classical Chinese      Image: Coptic    1    48K    Image: Sine Classic, Egyptian      Image: Croatian    1    199K    Image: Sine Classic      Image: Creatian    1    199K    Image: Sine Classic      Image: Creatian    1    199K    Image: Sine Classic    Image: Sine Classic      Image: Creatian    2    100K    Image: Sine Classic    Image: Sine Classic      Image: Dutch    2    306K    Image: Sine Classic    Image: Sine Classic    Image: Sine Classic      Image: Dutch		*)	Chinese	5	285K	2°ECV	Sino-Tibetan
Image: Coptic    1    48K    Image: Coptic    Afro-Asiatic, Egyptian      Image: Croatian    1    199K    Image: Coptic    IE, Slavic      Image: Croatian    5    2,227K    Image: Coptic	-	•	Chukchi	1	6K	P	Chukotko-Kamchatkan
Croatian    1    199K    IIQW    IE, Slavic      Czech    5    2,227K    IIQW    IE, Slavic      Danish    2    100K    IIII    IE, Germanic      Dutch    2    306K    IIV    IE, Germanic	-	gar.	Classical Chinese	1	233K	0	Sino-Tibetan
Czech    5    2,227K    Czech    IE, Slavic      Danish    2    100K    III (Germanic)      Dutch    2    306K    III (Germanic)	-	*	Coptic	1	48K	<b>4</b> 20	Afro-Asiatic, Egyptian
Danish    2    100K    IE, Germanic      Dutch    2    306K    IIIV    IE, Germanic	-		Croatian	1	199K	®₽₩	IE, Slavic
Dutch 2 306K 💷 V IE, Germanic	$\rightarrow$		Czech	5	2,227K	<b>₽&lt;⊅⊡0</b> ∆W	IE, Slavic
	$\rightarrow$		Danish	2	100K		IE, Germanic
▶ English 9 648K 含価型プロペ回の公司へのW IE, Germanic	$\rightarrow$		Dutch	2	306K	ew	IE, Germanic
	$\rightarrow$	X	English	9	648K	☞ⅲ록┛ፇ₢∢▣❶ሪᇒ⊃ଡ଼₩	IE, Germanic

#### https://universaldependencies.org/

## Universal Dependencies

### Train a classifier to predict transitions

- Given  $\{x_i, y_i\}$  where  $x_i$  is a sentence and  $y_i$  is a dependency parse
- generates  $y_i$ , so we can generate 2n training examples:  $\{(c_k, t_k)\}$ 
  - "shortest stack" strategy: prefer LEFT-ARC over SHIFT.
- The goal becomes how to learn a classifier from  $c_k$  to  $t_k$



• For each  $x_i$  with *n* words, we can construct a transition sequence of length 2n which  $c_k$ : configuration,  $t_k$ : transition

 $(2|\mathbf{R}| + 1)$  -way classification! R: dependency labels

### Train a classifier to predict transitions

During testing, we use the classifier to repeat predicting the transition, until we reach a terminal configuration

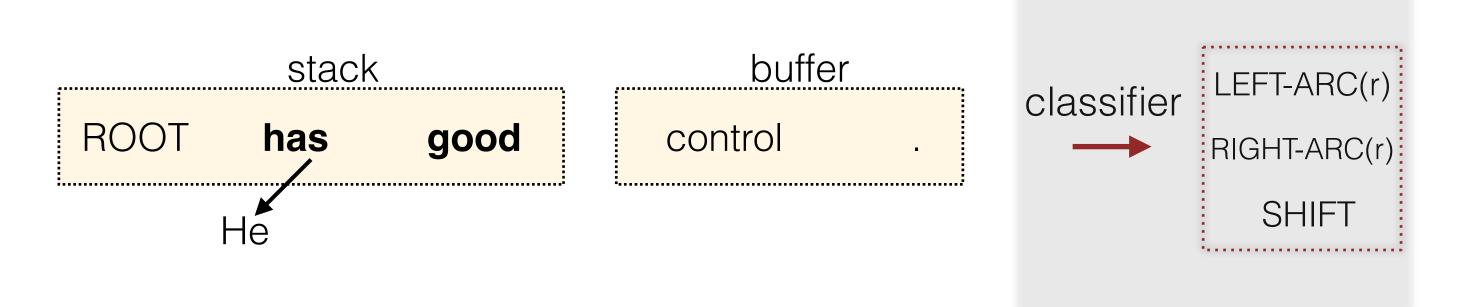
**function** DEPENDENCYPARSE(*words*) **returns** dependency tree

state  $\leftarrow$  {[root], [words], [] }; initial configuration while *state* not final

 $t \leftarrow Classifier (state)$ ; choose a transition operator to apply state  $\leftarrow$  APPLY(*t*, *state*); apply it, creating a new state

return *state* 

# Feature extraction



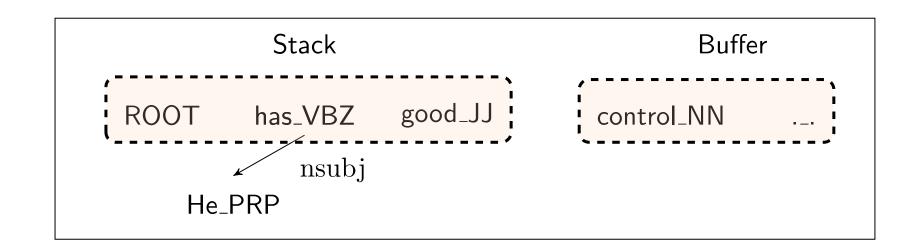
- Extract features from the configuration

Source	Feature templates		
One word	$s_1.w$	<i>s</i> <sub>1</sub> . <i>t</i>	$s_1.wt$
	$s_2.w$	<i>s</i> <sub>2</sub> . <i>t</i>	$s_2.wt$
	$b_1.w$	$b_1.w$	$b_0.wt$
Two word	$s_1.w \circ s_2.w$	$s_1.t \circ s_2.t$	$s_1.t \circ b_1.w$
	$s_1.t \circ s_2.wt$	$s_1.w \circ s_2.w \circ s_2.t$	$s_1.w \circ s_1.t \circ s_2.t$
	$s_1.w \circ s_1.t \circ s_2.t$	$s_1.w \circ s_1.t$	
			w: word, t: part-of-speech tag

(Nivre 2008): Algorithms for Deterministic Incremental Dependency Parsing

• Use your favorite classifier: logistic regression, SVM, FFNNs, ...

### Feature extraction



#### **Feature templates**

$$s_2 \cdot w \circ s_2 \cdot t$$
$$s_1 \cdot w \circ s_1 \cdot t \circ b_1 \cdot w$$

#### Today we can use neural networks to extract features!

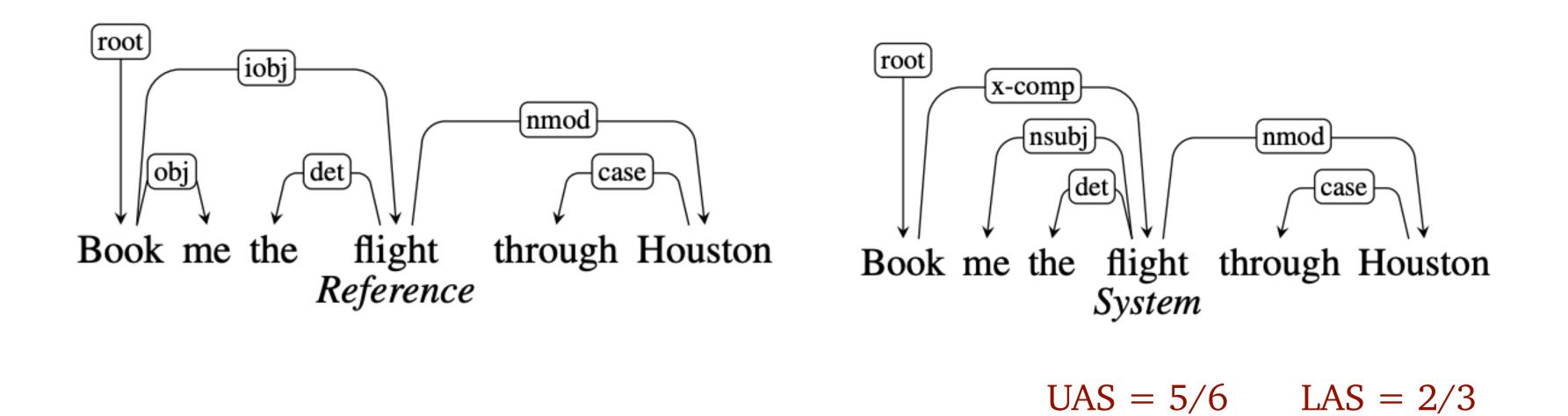
(Nivre 2008): Algorithms for Deterministic Incremental Dependency Parsing

w: words, t: part-of-speech tags

### Features $s_2 \cdot w = has \circ s_2 \cdot t = VBZ$ $s_1 \cdot w = good \circ s_1 \cdot t = JJ \circ b_1 \cdot w = control$

# Evaluating dependency parsing

- Unlabeled attachment score (UAS)
- Labeled attachment score (LAS)



= percentage of words that have been assigned the correct head

= percentage of words that have been assigned the correct head & label

## Evaluating dependency parsing

#### Parser

(Chen and Manning, 2014) (Dyer et al., 2015) (Ballesteros et al., 2016) (Weiss et al., 2015) (Andor et al., 2016) (Ma et al., 2018) §

(Kiperwasser and Goldberg (Kiperwasser and Goldberg (Wang and Chang, 2016) (Cheng et al., 2016) (Kuncoro et al., 2016) (Zheng, 2017) § (Dozat and Manning, 2017)

T: transition-based / G: graph-based

		Test		
		UAS	LAS	
)	T	91.8 93.1 93.56 94.26 94.61 95.87	89.6 90.9 92.41 91.42 92.79 94.19	
rg, 2016a) § rg, 2016b)	G	93.0 93.1 94.08 94.10 94.26 95.53 95.74	90.9 91.0 91.82 91.49 92.06 93.94 94.08	