Precept 5: Parsing

Howard Chen
03/03/2023

Agenda

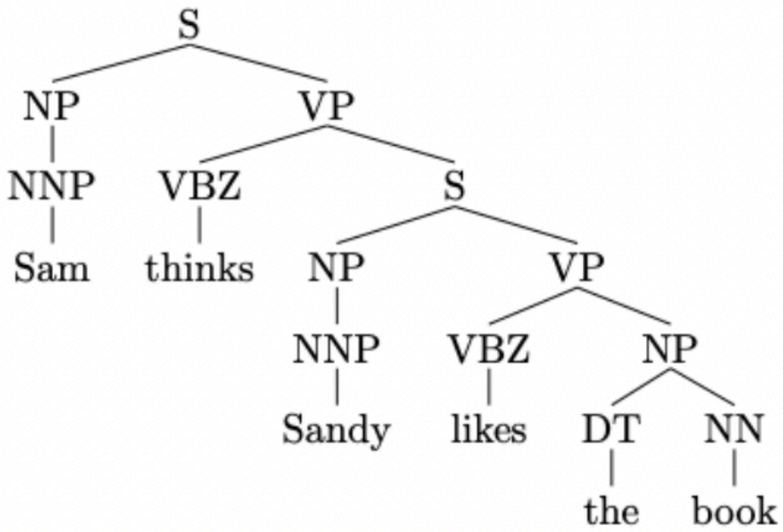
- Constituency Parsing
- Dependency Parsing
- Questions

Constituency Parsing

Input

Sam thinks Sandy likes the book

Output



Constituency Parsing

- Probabilistic Context-Free Grammars (PCFGs)

Defines a bunch of production rules.

Constituency Parsing

- Probabilistic Context-Free Grammars (PCFGs)
- MLE for learning

$R, q =$

S	→	NP	VP	1.0
VP	→	Vi		0.3
VP	→	Vt	NP	0.5
VP	→	VP	PP	0.2
NP	→	DT	NN	0.8
NP	→	NP	PP	0.2
PP	→	IN	NP	1.0

Vi	→	sleeps	1.0
Vt	→	saw	1.0
NN	→	man	0.1
NN	→	woman	0.1
NN	→	telescope	0.3
NN	→	dog	0.5
DT	→	the	1.0
IN	→	with	0.6
IN	→	in	0.4

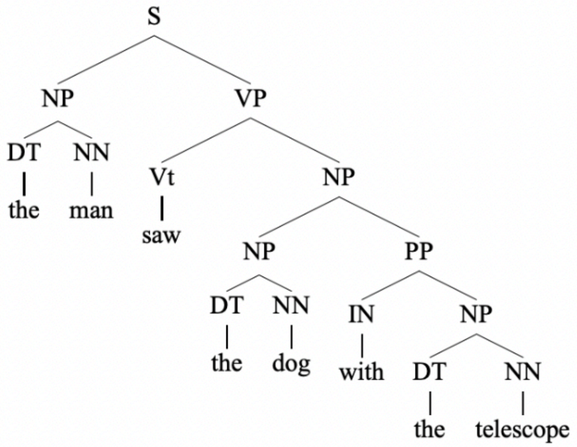
Constituency Parsing

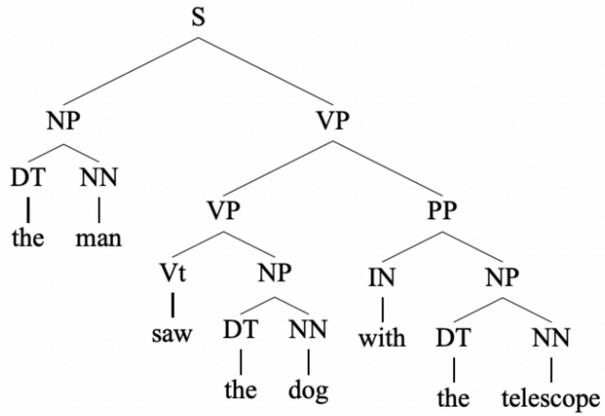
- Probabilistic Context-Free Grammars (PCFGs)
- MLE for learning
- Parsing PCFGs: the CKY algorithm

$R, q =$

S	→	NP	VP	1.0
VP	→	Vi		0.3
VP	→	Vt	NP	0.5
VP	→	VP	PP	0.2
NP	→	DT	NN	0.8
NP	→	NP	PP	0.2
PP	→	IN	NP	1.0

Vi	→	sleeps	1.0
Vt	→	saw	1.0
NN	→	man	0.1
NN	→	woman	0.1
NN	→	telescope	0.3
NN	→	dog	0.5
DT	→	the	1.0
IN	→	with	0.6
IN	→	in	0.4





$$q(\text{VP} \rightarrow \text{Vt}\ \ \text{NP}) \times q(\text{NP} \rightarrow \text{NP}\ \ \text{PP}) = 0.5 \times 0.2 = 0.1$$

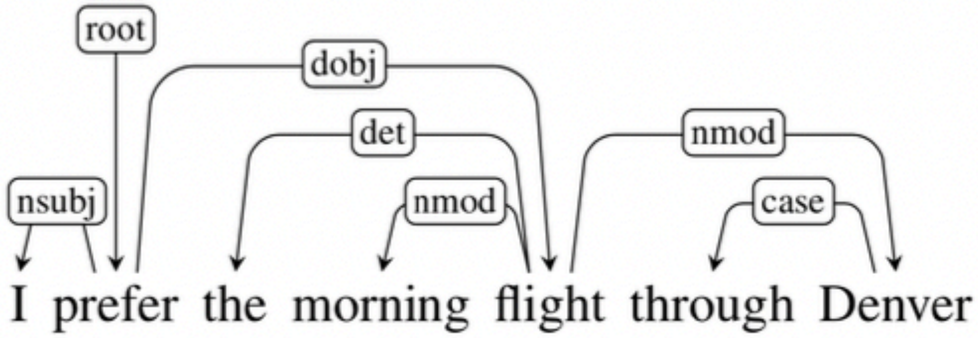
$$q(\text{VP} \rightarrow \text{VP}\ \ \text{PP}) \times q(\text{VP} \rightarrow \text{Vt}\ \ \text{NP}) = 0.2 \times 0.5 = 0.1$$

Dependency Parsing

Input

I prefer the morning flight through Denver

Output



Dependency Parsing

- **Arc-standard algorithm**

Dependency Parsing

- **Arc-standard algorithm**

How does a dependency parser parse a sentence into a tree?

Dependency Parsing

- **Arc-standard algorithm**
- Input: a sequence of words
- Output: a parse (i.e., which word points to which word)

Dependency Parsing

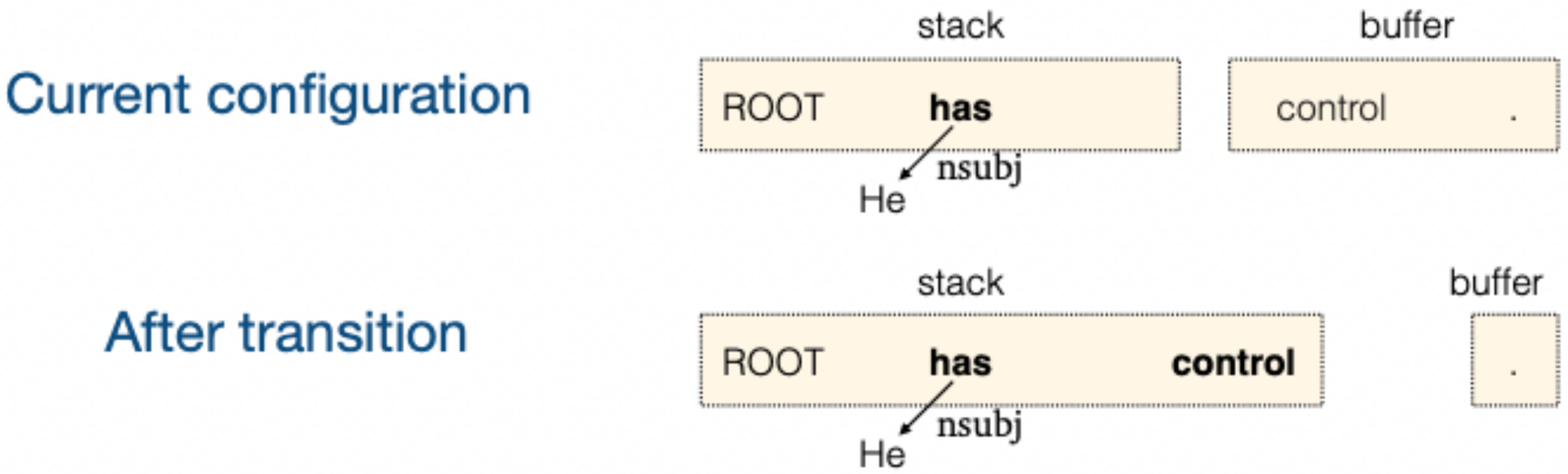
- **Arc-standard algorithm**
- Input: a sequence of words
- Output: a parse (i.e., which word points to which word)
- Configuration: stack (s) + buffer (b) + a set of arcs (A)
- Starting config: $s = [\text{ROOT}], b = [w_1, w_2, \ldots, w_n], A = \emptyset$
- End config: $s = [\text{ROOT}], b = \emptyset$

Dependency Parsing

- **Arc-standard algorithm**
- Input: a sequence of words
- Output: a parse (i.e., which word points to which word)
- Configuration: stack (s) + buffer (b) + a set of arcs (A)
- Starting config: $s = [\text{ROOT}], b = [w_1, w_2, \ldots, w_n], A = \emptyset$
- End config: $s = [\text{ROOT}], b = \emptyset$
- Actions: SHIFT, LEFT-ARC(r), RIGHT-ARC(r)

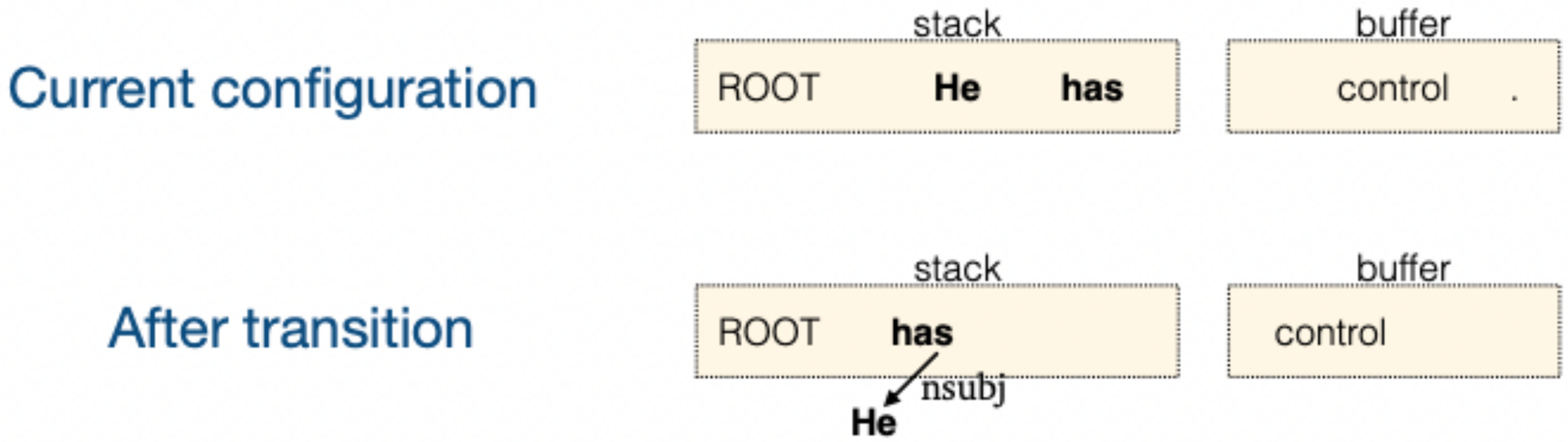
Dependency Parsing

- SHIFT: move word at the top of buffer -> top of stack



Dependency Parsing

- SHIFT: move word at the top of buffer -> top of stack
- LEFT-ARC(r)

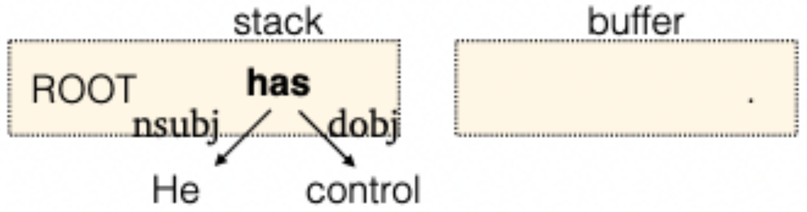


Dependency Parsing

- SHIFT: move word at the top of buffer -> top of stack
- LEFT-ARC(r)
- RIGHT-ARC(r)

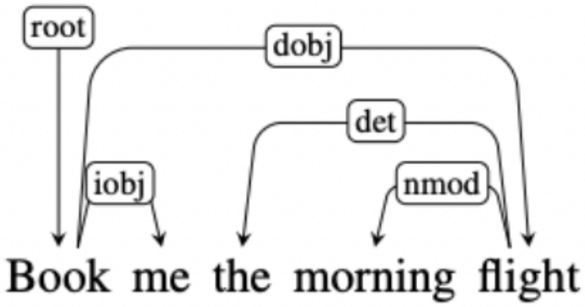
Current configuration

After transition

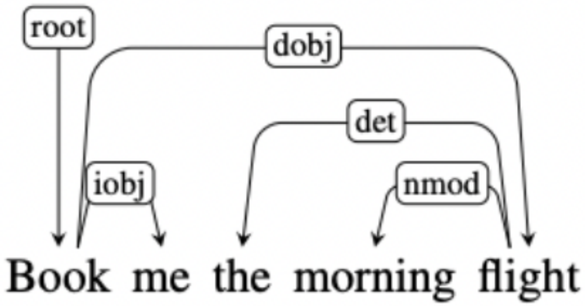


Dependency Parsing

	stack	buffer	action	added arc
0	[ROOT]	[Book, me, the, morning, flight]	SHIFT	

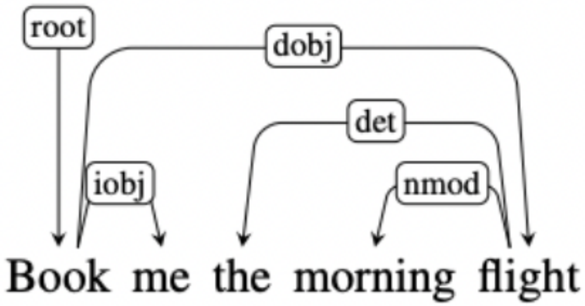


Dependency Parsing



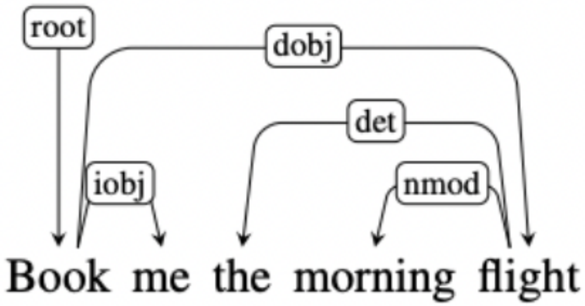
	stack	buffer	action	added arc
0	[ROOT]	[Book, me, the, morning, flight]	SHIFT	
1	[ROOT, Book]	[me, the, morning, flight]	SHIFT	

Dependency Parsing



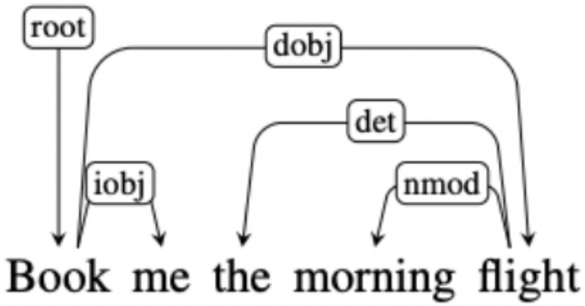
	stack	buffer	action	added arc
0	[ROOT]	[Book, me, the, morning, flight]	SHIFT	
1	[ROOT, Book]	[me, the, morning, flight]	SHIFT	
2	[ROOT, Book, me]	[the, morning, flight]	RIGHT-ARC(iobj)	(Book, iobj, me)

Dependency Parsing



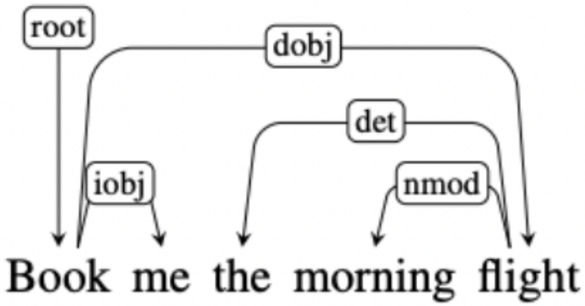
	stack	buffer	action	added arc
0	[ROOT]	[Book, me, the, morning, flight]	SHIFT	
1	[ROOT, Book]	[me, the, morning, flight]	SHIFT	
2	[ROOT, Book, me]	[the, morning, flight]	RIGHT-ARC(iobj)	(Book, iobj, me)
3	[ROOT, Book]	[the, morning, flight]	SHIFT	

Dependency Parsing



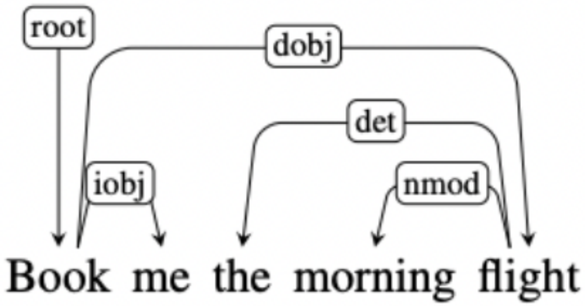
	stack	buffer	action	added arc
0	[ROOT]	[Book, me, the, morning, flight]	SHIFT	
1	[ROOT, Book]	[me, the, morning, flight]	SHIFT	
2	[ROOT, Book, me]	[the, morning, flight]	RIGHT-ARC(iobj)	(Book, iobj, me)
3	[ROOT, Book]	[the, morning, flight]	SHIFT	
4	[ROOT, Book, the]	[morning, flight]	SHIFT	
5	[ROOT, Book, the, morning]	[flight]	SHIFT	

Dependency Parsing



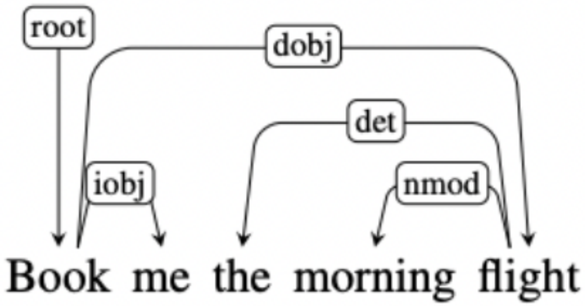
	stack	buffer	action	added arc
0	[ROOT]	[Book, me, the, morning, flight]	SHIFT	
1	[ROOT, Book]	[me, the, morning, flight]	SHIFT	
2	[ROOT, Book, me]	[the, morning, flight]	RIGHT-ARC(iobj)	(Book, iobj, me)
3	[ROOT, Book]	[the, morning, flight]	SHIFT	
4	[ROOT, Book, the]	[morning, flight]	SHIFT	
5	[ROOT,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

Dependency Parsing



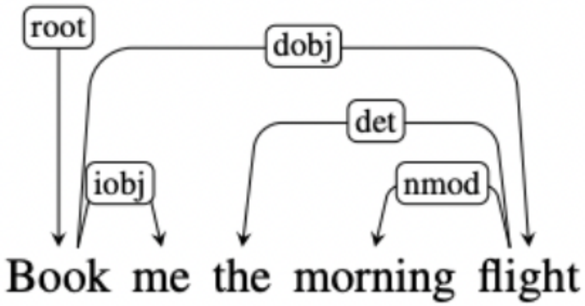
	stack	buffer	action	added arc
0	[ROOT]	[Book, me, the, morning, flight]	SHIFT	
1	[ROOT, Book]	[me, the, morning, flight]	SHIFT	
2	[ROOT, Book, me]	[the, morning, flight]	RIGHT-ARC(iobj)	(Book, iobj, me)
3	[ROOT, Book]	[the, morning, flight]	SHIFT	
4	[ROOT, Book, the]	[morning, flight]	SHIFT	
5	[ROOT, 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)

Dependency Parsing



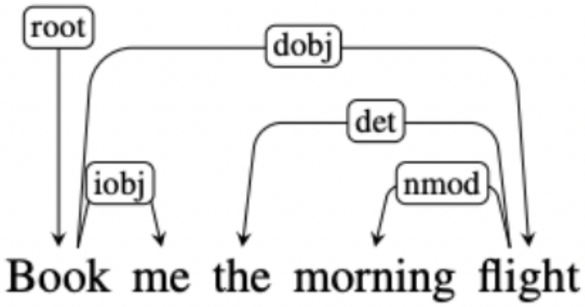
	stack	buffer	action	added arc
0	[ROOT]	[Book, me, the, morning, flight]	SHIFT	
1	[ROOT, Book]	[me, the, morning, flight]	SHIFT	
2	[ROOT, Book, me]	[the, morning, flight]	RIGHT-ARC(iobj)	(Book, iobj, me)
3	[ROOT, Book]	[the, morning, flight]	SHIFT	
4	[ROOT, Book, the]	[morning, flight]	SHIFT	
5	[ROOT, 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)	(Book,dobj,flight)

Dependency Parsing



	stack	buffer	action	added arc
0	[ROOT]	[Book, me, the, morning, flight]	SHIFT	
1	[ROOT, Book]	[me, the, morning, flight]	SHIFT	
2	[ROOT, Book, me]	[the, morning, flight]	RIGHT-ARC(iobj)	(Book, iobj, me)
3	[ROOT, Book]	[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)	(Book,dobj,flight)
9	[ROOT, Book]	[]	RIGHT-ARC(root)	(ROOT,root,Book)

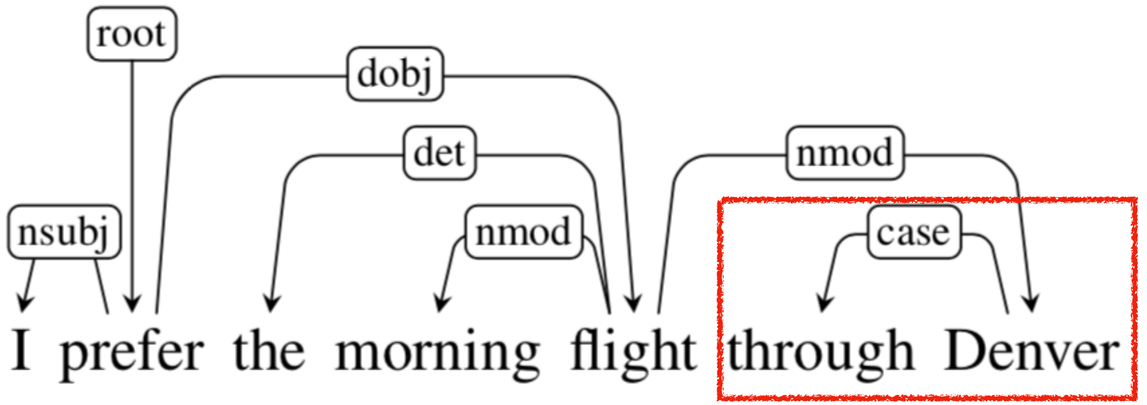
Dependency Parsing



	stack	buffer	action	added arc
0	[ROOT]	[Book, me, the, morning, flight]	SHIFT	
1	[ROOT, Book]	[me, the, morning, flight]	SHIFT	
2	[ROOT, Book, me]	[the, morning, flight]	RIGHT-ARC(iobj)	(Book, iobj, me)
3	[ROOT, Book]	[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)	(Book,dobj,flight)
9	[ROOT, Book]	[]	RIGHT-ARC(root)	(ROOT,root,Book)
10	[ROOT]	[]		

Dependency Parsing

- Common confusion: the ‘case’ label points into the preposition word



Dependency Parsing

- Where does the dependency parser come from?

Dependency Parsing

- Where does the dependency parser come from?
- We need to train it using statistical learning methods!

Dependency Parsing

- Where does the dependency parser come from?
- We need to train it using statistical learning methods!
- Collect datasets $\{(x_i, y_i)\}_{i=1}^{N}$
- x: sentence
- y: dependency parse tree

Dependency Parsing

- Where does the dependency parser come from?
- We need to train it using statistical learning methods!
- Collect datasets $\{(x_i, y_i)\}_{i=1}^{N}$
- x: sentence
- y: dependency parse tree
- How do we use annotated these data to train a parser?

Dependency Parsing

- Model needs to learn: $c \rightarrow a$ (configuration to action mapping)

$$a = M(c)$$

Dependency Parsing

- Model needs to learn: $c \rightarrow a$ (configuration to action mapping)
- We can generate this from the annotated $\{(x_i, y_i)\}_{i=1}^{N}$

Dependency Parsing

- Model needs to learn: $c \rightarrow a$ (configuration to action mapping)
- We can generate this from the annotated $\{(x_i, y_i)\}_{i=1}^{N}$
- Run arc-standard, we can collect $2n$ (c, a) pairs from one (x, y) pair, where n is the number of words in x

Dependency Parsing

- Model needs to learn: $c \rightarrow a$ (configuration to action mapping)
- We can generate this from the annotated $\{(x_i, y_i)\}_{i=1}^{N}$
- Run arc-standard, we can collect $2n$ (c, a) pairs from one (x, y) pair, where n is the number of words in x
- To complete the process, each word needs to be shifted and merged exactly once

Dependency Parsing

- Model needs to learn: $c \rightarrow a$ (configuration to action mapping)
- We can generate this from the annotated $\{(x_i, y_i)\}_{i=1}^{N}$
- Run arc-standard, we can collect $2n$ (c, a) pairs from one (x, y) pair, where n is the number of words in x
- To complete the process, each word needs to be shifted and merged exactly once
- Number of actions: $2|R| + 1$ (+SHIFT)

Dependency Parsing

- **Evaluation**
- Unlabeled attachment score (UAS): only look at the arcs
- Labeled attachment score (LAS): look at both the arcs and the labels

Questions?