Ling/CSE 472: Introduction to Computational Linguistics

5/7: Dependency parsing

Overview

- Grammatical dependencies
- Dependency grammar
- Dependency treebanks
- Dependency parsing
- Reading questions (with headers)
- Questions about milestone 2

Grammatical Dependencies

- Relate words in the sentence to each other
- A labeled with the type of dependency
- Are typically represented as graphs (sometimes trees)
  - Where each node is a word in the sentence
  - Where word in the sentence is (usually) a node

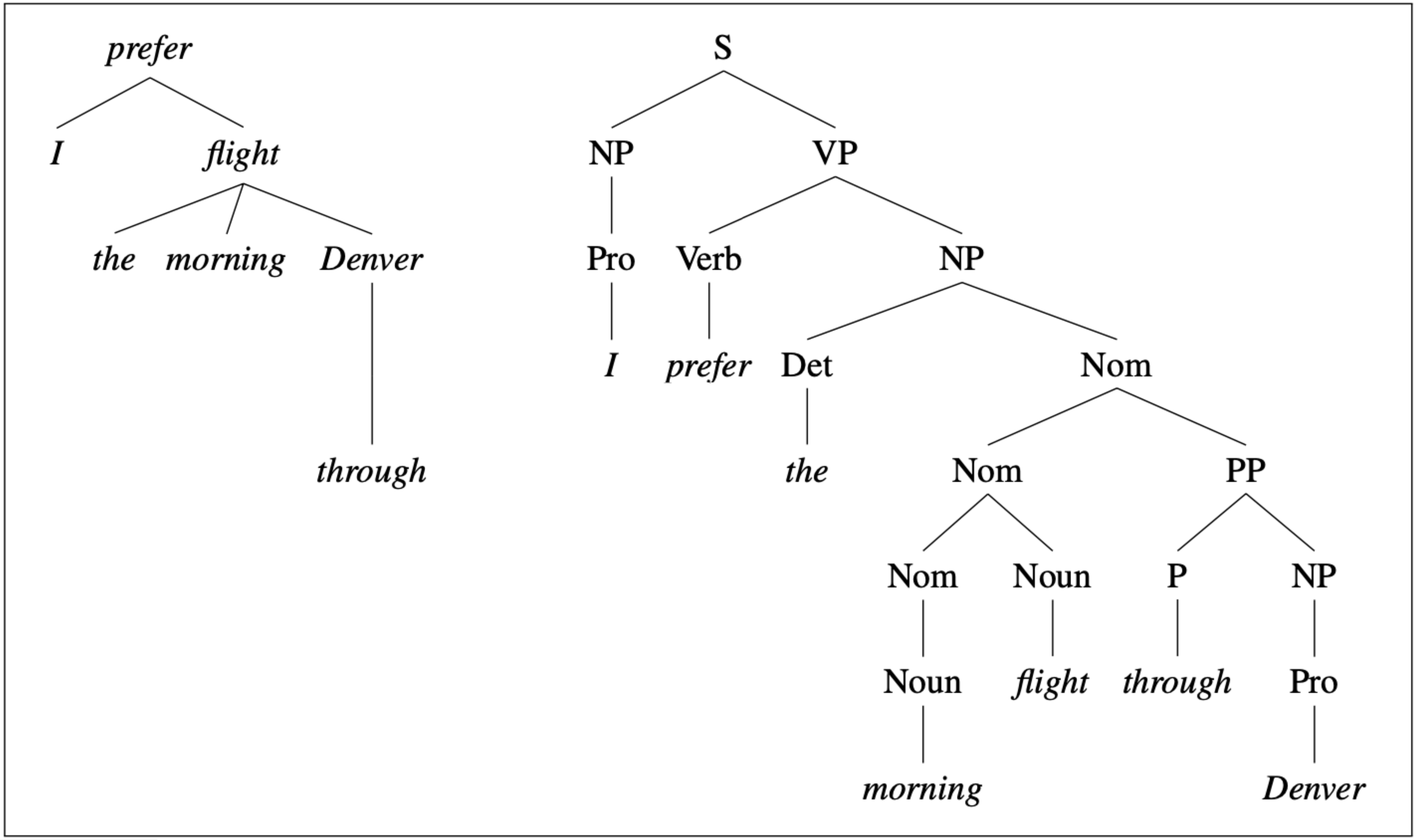
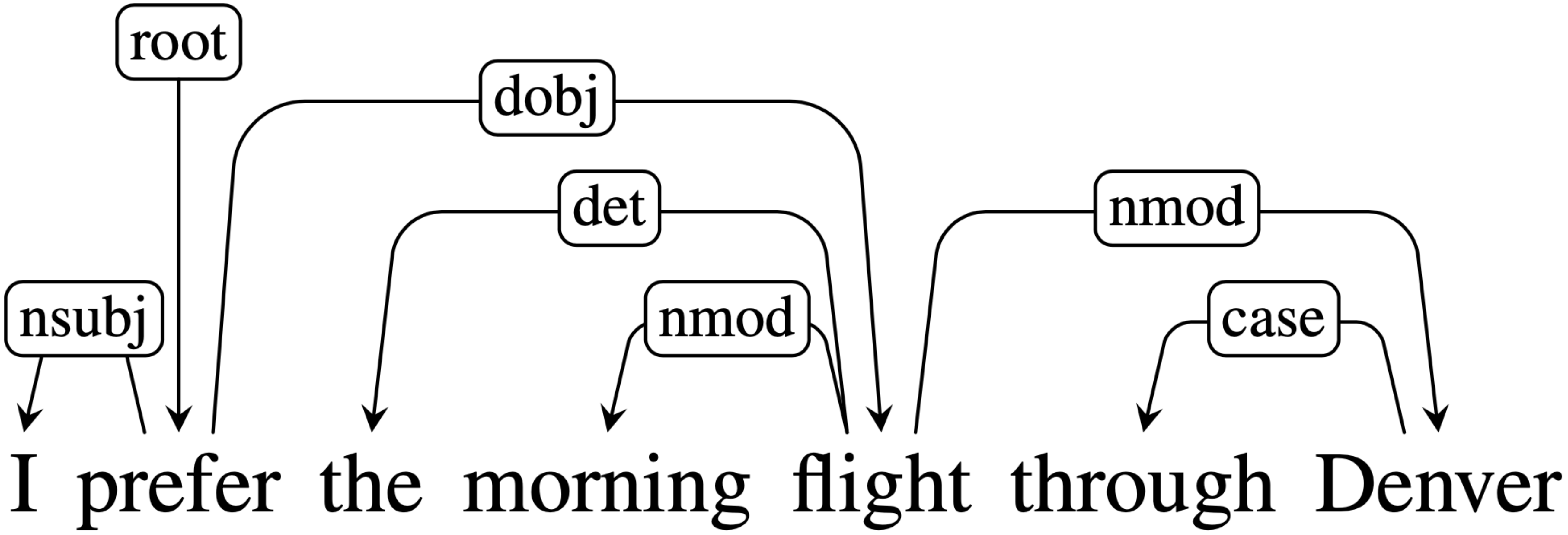


Figure 15.1 A dependency-style parse alongside the corresponding constituent-based analysis for *I prefer the morning flight through Denver.*


root
dobj
det
nmod
nsubj
nmod
case
I prefer the morning flight through Denver

Dependency Grammar

- Theoretical foundations: Tesnière 1959, Mel’čuk 1988, Hudson 1984, Sgall et al 1986
- Focus not on grammaticality (“What’s a possible sentence?”) but on grammatical structure, given a string

Dependency Treebanks: Universal Dependencies

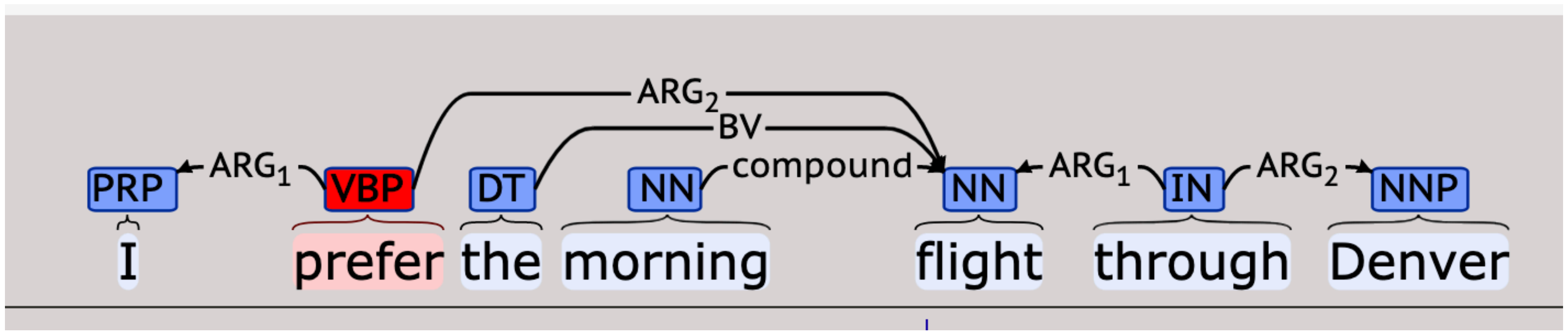
- https://universaldependencies.org/
- Builds on:
  - Stanford dependencies (LFG-inspired transformation of CFG representations for English from the Stanford parser)
  - Theoretical work on dependency grammar
  - “Universal” POS tagset developed initially for cross-linguistic error analysis (McDonald and Nivre 2007)

What is needed for UD to be successful? (from universaldependencies.org/introduction.html)

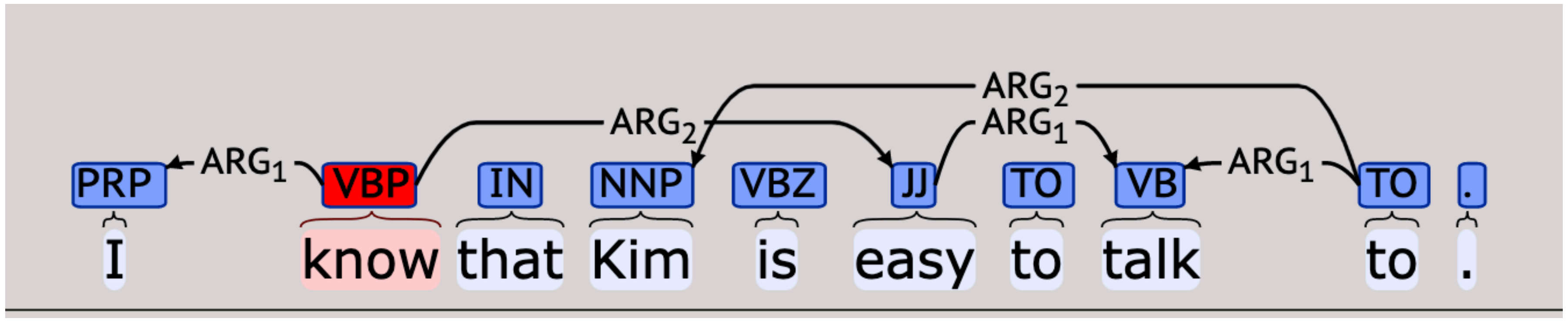
- The secret to understanding the design and current success of UD is to realize that the design is a very subtle compromise between approximately 6 things:
    - UD needs to be satisfactory on linguistic analysis grounds for individual languages.
    - UD needs to be good for linguistic typology, i.e., providing a suitable basis for bringing out cross-linguistic parallelism across languages and language families.
    - UD must be suitable for rapid, consistent annotation by a human annotator.
    - UD must be suitable for computer parsing with high accuracy.
    - UD must be easily comprehended and used by a non-linguist, whether a language learner or an engineer with prosaic needs for language processing. We refer to this as seeking a habitable design, and it leads us to favor traditional grammar notions and terminology.
    - UD must support well downstream language understanding tasks (relation extraction, reading comprehension, machine translation, …).
- It’s easy to come up with a proposal that improves UD on one of these dimensions. The interesting and difficult part is to improve UD while remaining sensitive to all these dimensions.

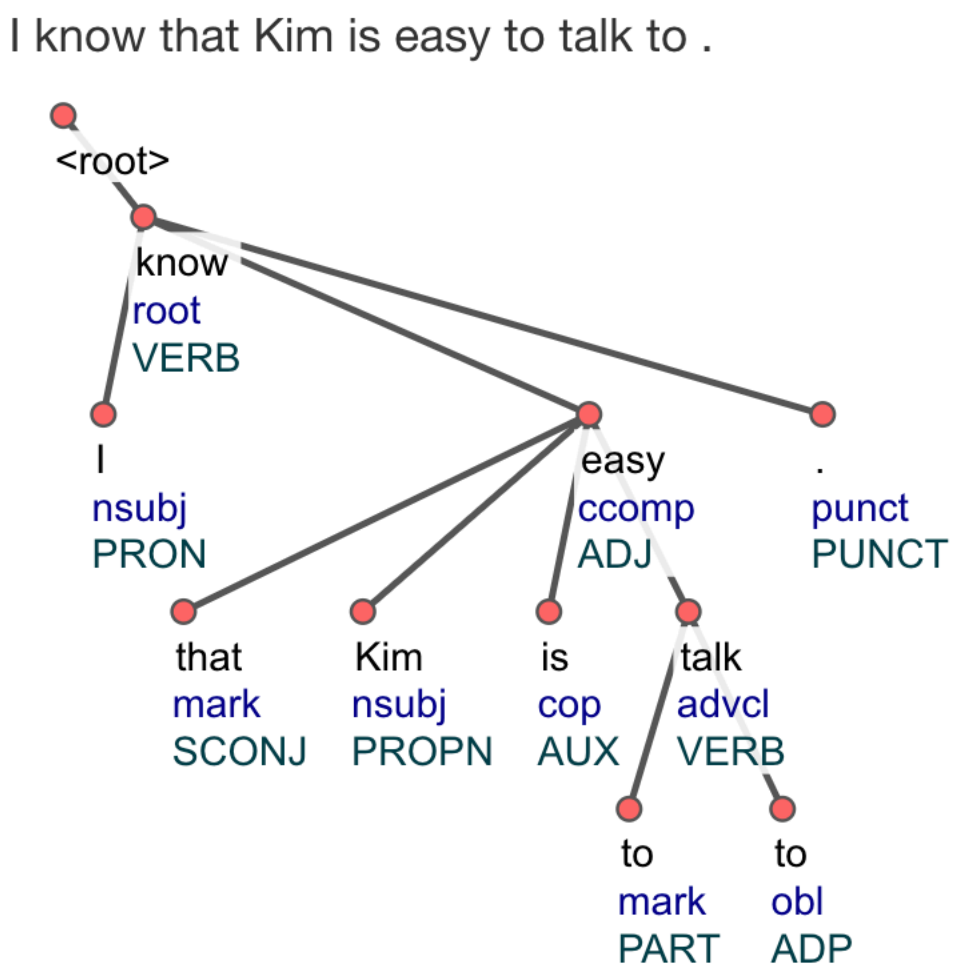
Dependency Treebanks outside UD

- Richer grammatical formalisms such as HPSG can be 'boiled down' to dependency representations
    - Syntactic OR semantic dependencies (Ivanova et al 2012)



DM v. UD


PRP
VBP
IN
NNP
VBZ
JJ
TO
VB
TO
.
ARG1
ARG2
ARG2
ARG1
ARG1
I know that Kim is easy to talk to .


I know that Kim is easy to talk to .
<root>
know root VERB
I nsubj PRON
easy ccomp ADJ
. punct PUNCT
that mark SCONJ
Kim nsubj PROPN
is cop AUX
talk advcl VERB
to mark PART
to obl ADP

Dependency Parsing

- Transition-based v. graph-based
- Feature templates v. neural
- Source of training data

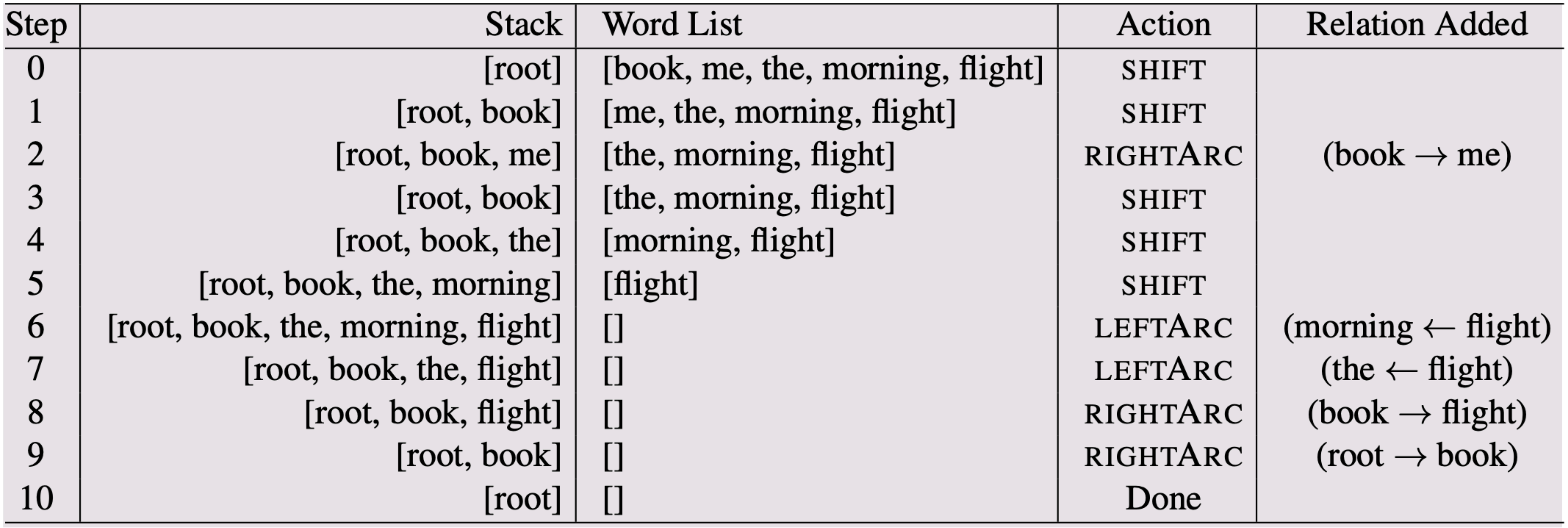
Transition-Based Dependency Parsing

```
function DEPENDENCYPARSE(words) returns dependency tree

  state ← {[root], [words], [] }  ; initial configuration
  while state not final
    t ← ORACLE(state)         ; choose a transition operator to apply
    state ← APPLY(t, state)  ; apply it, creating a new state
  return state
```

Figure 15.6 A generic transition-based dependency parser

Transition-Based Dependency Parsing



Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book $\rightarrow$ 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]	[]	LEFTARC	(morning $\leftarrow$ flight)
7	[root, book, the, flight]	[]	LEFTARC	(the $\leftarrow$ flight)
8	[root, book, flight]	[]	RIGHTARC	(book $\rightarrow$ flight)
9	[root, book]	[]	RIGHTARC	(root $\rightarrow$ book)
10	[root]	[]	Done	

Figure 15.7 Trace of a transition-based parse.

Learning transition scores

$$\langle s_1.w = \textit{flights}, op = \textit{shift}\rangle \quad (18.11)$$
$$\langle s_2.w = \textit{canceled}, op = \textit{shift}\rangle$$
$$\langle s_1.t = NNS, op = \textit{shift}\rangle$$
$$\langle s_2.t = VBD, op = \textit{shift}\rangle$$
$$\langle b_1.w = \textit{to},$$
$$\langle b_1.t = TO,$$
$$\langle s_1.wt = \textit{flightsNNS},$$

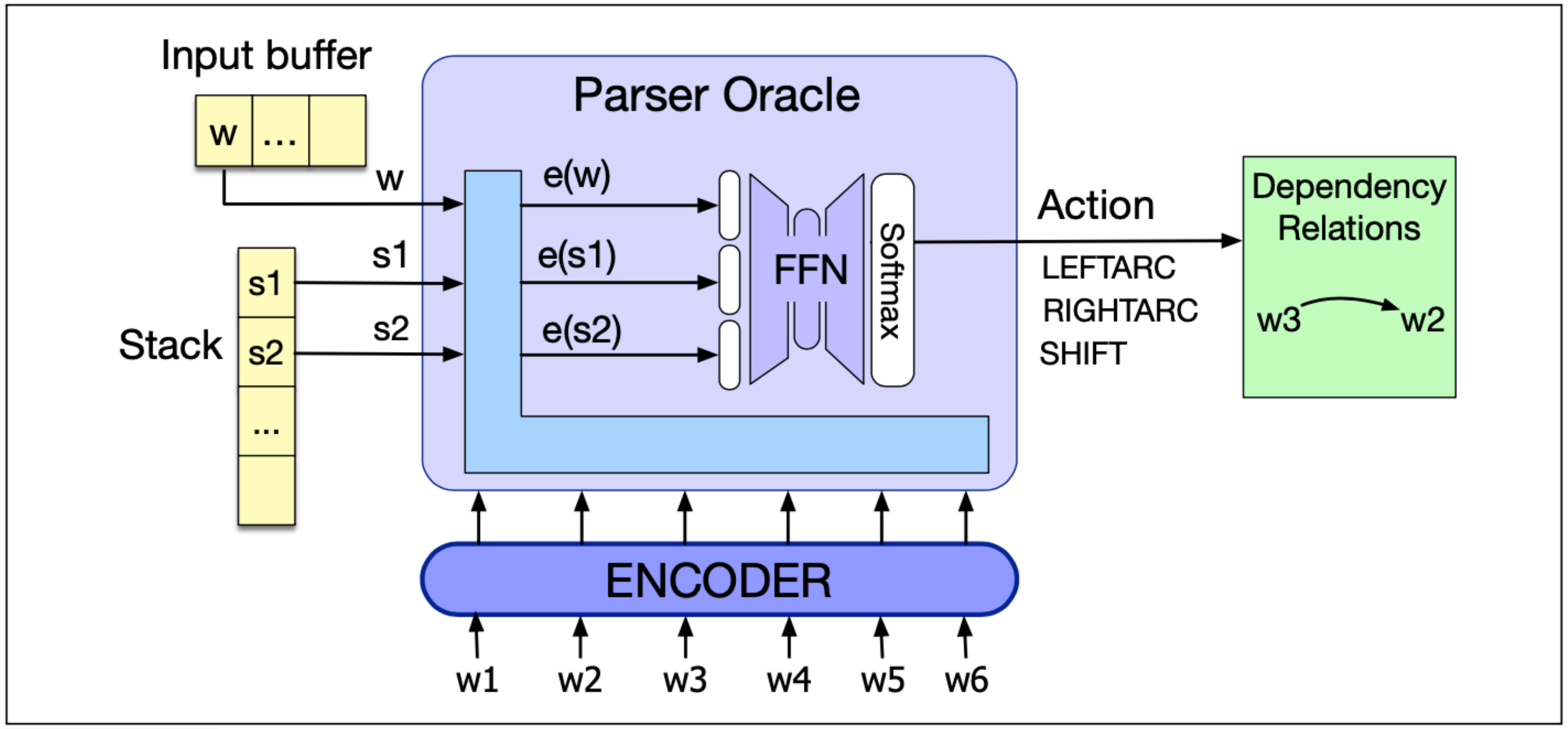


Figure 18.8 Neural classifier for the oracle for the transition-based parser. The parser takes the top 2 words on the stack and the first word of the buffer, represents them by their encodings (from running the whole sentence through the encoder), concatenates the embeddings and passes through a softmax to choose a parser action (transition).

Graph-based parsing

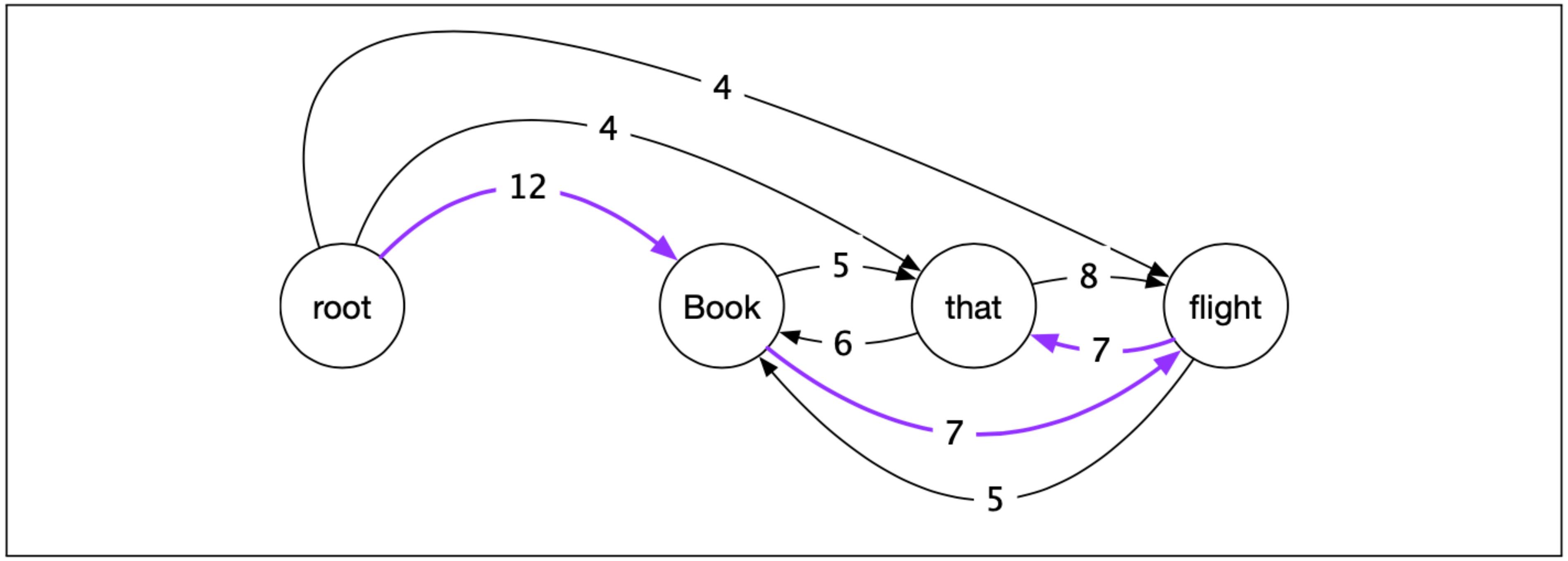


Figure 18.11 Initial rooted, directed graph for *Book that flight*.

Reading questions

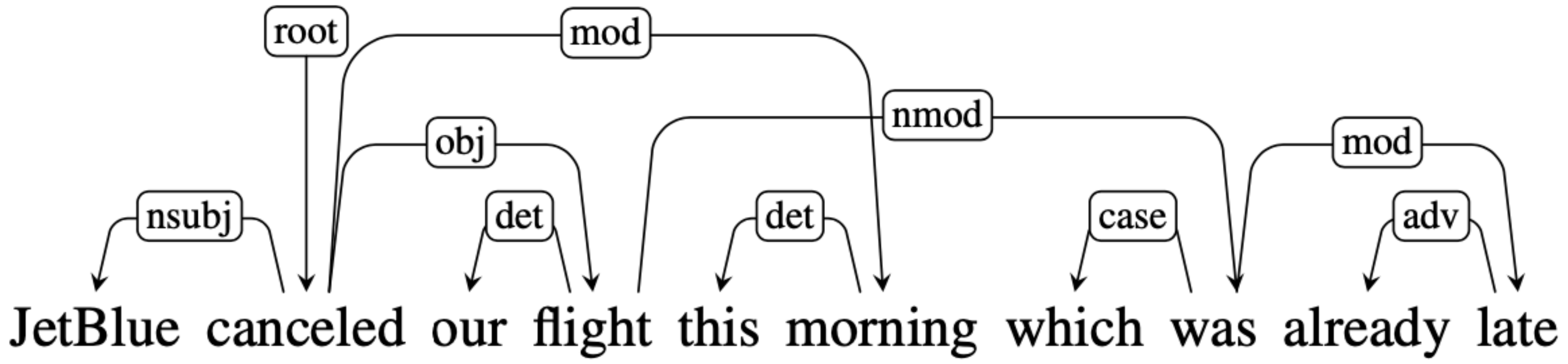
- Is a good way of thinking of Projectivity to compare it to the syntax notion of dominance? (I think that's the right term but the notion that a sister of a head dominates everything it's other sister dominates? If that's even how it works - it's been a while)

- Projectivity is kind of hard for me to conceptualize, at least in the way diagrams are presented in this chapter. I'm more familiar with traditional syntax trees than dependency parses, so what would the equivalent of projectivity be in a syntax tree? Is it like c-commanding, or some other idea entirely that isn't discussed in typical syntax?

- The concept of projective and non-projective arcs/trees: How exactly do they differ? Also, is it possible to visualize the distinction with typical syntax trees?

Non-projectivity

- J&M: “An arc from a head to a dependent is said to be projective if there is a path from the head to every word that lies between the head and the dependent in the sentence.” (Ch 18, p.4)


root
mod
obj
nmod
mod
nsubj
det
det
case
adv
JetBlue canceled our flight this morning which was already late

(18.3)

Reading questions

- Are sentences themselves inherently projective/non-projective, or does it depend on how the tree is drawn?
- The book mentions that widely used English treebanks always generate projective trees. Does this imply the same sentence could be used to construct both projective and non-projective trees?

Reading questions

- Is there a direct relationship between the level of flexibility of a language's word order vs if its trees are projective or not? For example, can projective trees ever be used to represent languages with flexible word order?

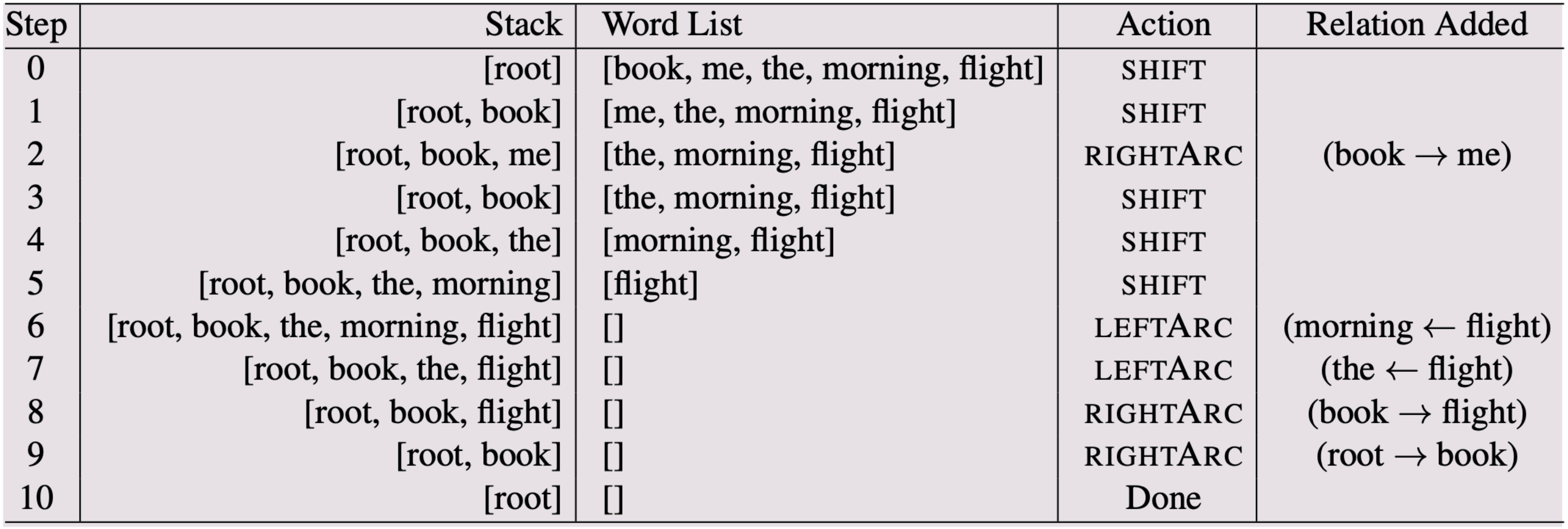
- Is dependency parsing or constituency parsing better at handling ambiguity?

Reading questions

- Which word is the head in the LEFTARC vs RIGHTARC transitions? "LEFTARC: Assert a head-dependent relation between the word at the top of the stack and the second word; remove the second word from the stack." Does this mean that the word at the top is the head and we remove the second, dependent word?

- Also, when we "postpone dealing with the current word, storing it for later processing," when do we come back it? How is the dependency relationship assigned thereafter?

Dependency Parsing



Step	Stack	Word List	Action	Relation Added
0	[root]	[book, me, the, morning, flight]	SHIFT	
1	[root, book]	[me, the, morning, flight]	SHIFT	
2	[root, book, me]	[the, morning, flight]	RIGHTARC	(book → 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]	[]	LEFTARC	(morning ← flight)
7	[root, book, the, flight]	[]	LEFTARC	(the ← flight)
8	[root, book, flight]	[]	RIGHTARC	(book → flight)
9	[root, book]	[]	RIGHTARC	(root → book)
10	[root]	[]	Done	

Figure 15.7 Trace of a transition-based parse.

Reading questions

- What exactly do the edge scores for graph-based parsing represent? Do they represent how "correct" a certain dependency is?

Reading questions

- In the graph-based parsing, we have a way to assign scores between words and we have ways to use these scores to decide which one to included. However, the non-greedy version of the transition-based parsing, i.e. the version it can compare between different parses does not have a clear way to get the score matrix from? Can we use method described in graph-based parsing to do that? If not, is there developed method on computing these scores using the principle of transition-based parsing?

Reading questions

- Which method of showing dependency parcing is more widely used? Because at least to me, I feel like I have a better understanding of how the words are linked together in the normal sentence forms with the arcs rather than that graph with just a bunch of lines pointing to things.

Reading questions

- The textbook mentions that graph-based dependency parsing often utilizes the Cho and Liu and Edmonds algorithm to find the maximum spanning tree, which runs with a time complexity of O(mn) - equivalent to O(n3). Would it be possible to perform this same operation with the more common Kruskal's MST algorithm, which instead runs in O(m log(m)) time? While Kruskal's finds minimum spanning trees, we could potentially negate the edge weights to find the maximum tree.

Milestone 2, due 5/12

- Submit an update on how your project is doing, and what needs to change from your original plan to be successful clear. This update should include:
  - A detailed description of project, according to the M2 specification for your selected project type.
  - A description what needs to change from your original proposal, and a rationale.

NLP/compling in the news

- https://www.theguardian.com/technology/2023/may/04/bernie-sanders-elon-musk-and-white-house-seeking-my-help-says-godfather-of-ai

- https://www.vice.com/en/article/wxjdg5/scary-emergent-ai-abilities-are-just-a-mirage-produced-by-researchers-stanford-study-says