Language Technology (2021)

Syntactic analysis

Marco Kuhlmann
Department of Computer and Information Science

LINKÖPING UNIVERSITY

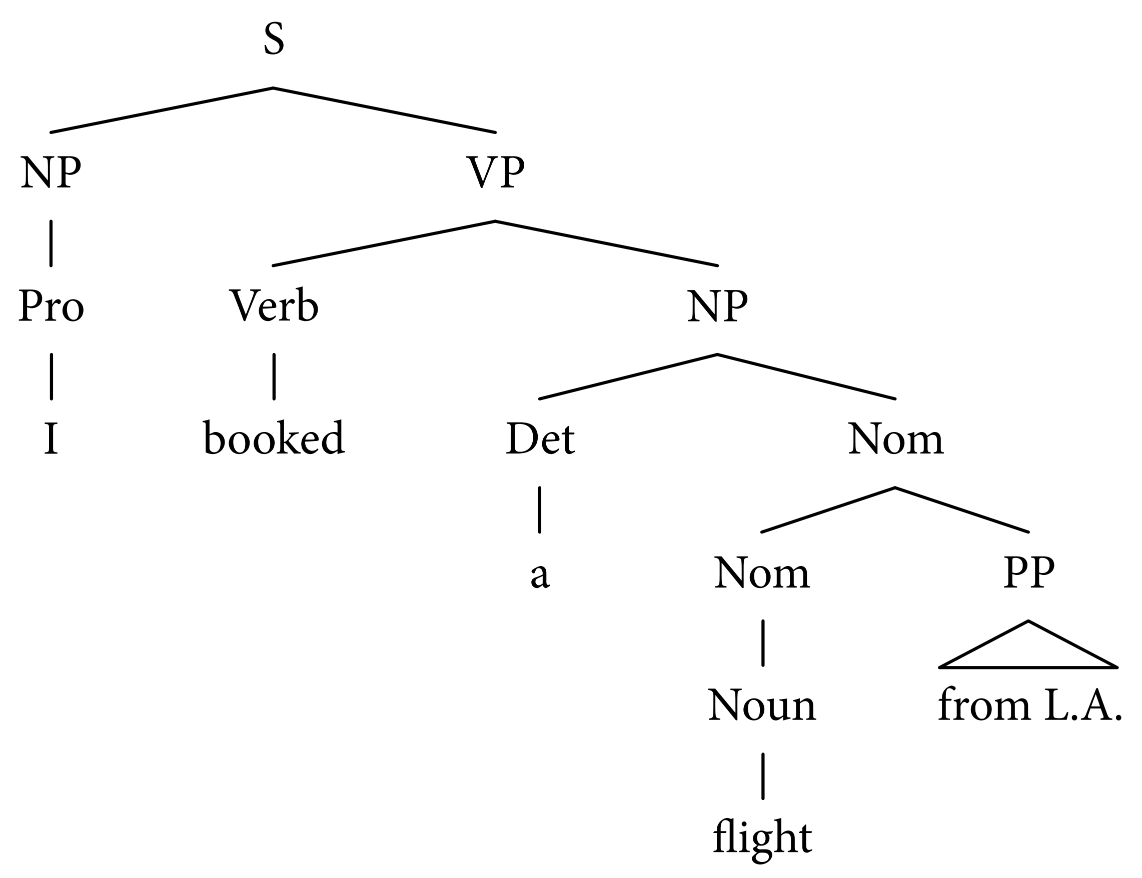
Syntactic analysis

- **Syntactic analysis** or **syntactic parsing** is the task to map a sentence to a formal representation of its syntactic structure.
- The syntactic structure of a sentence provides important clues about the meaning of the sentence.

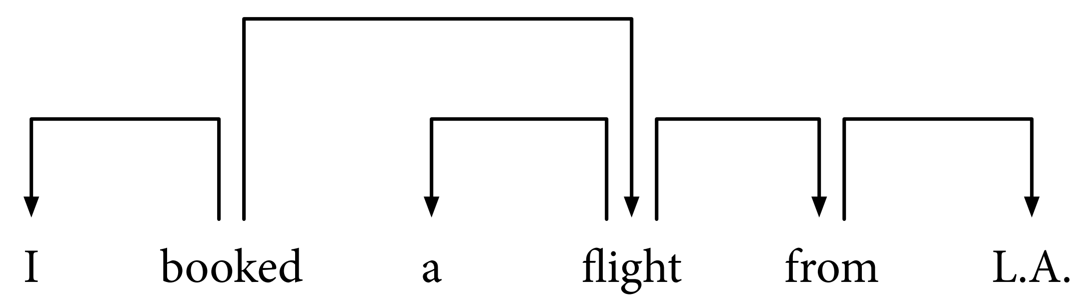
  example application: information extraction

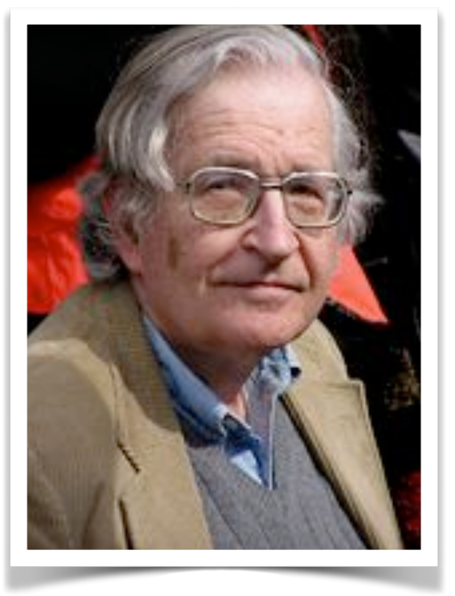
Different syntactic representations

Phrase structure tree

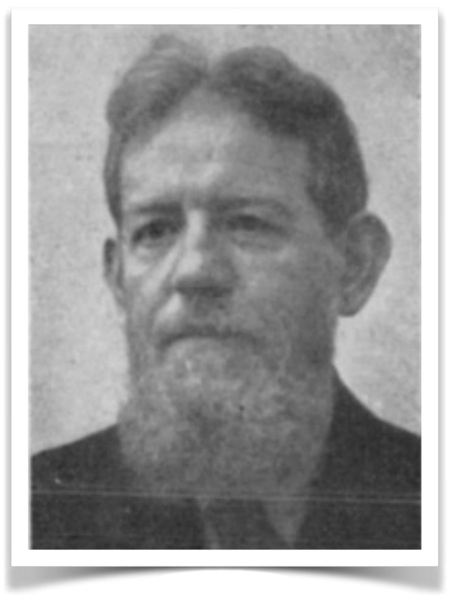


Dependency tree





Noam Chomsky



Lucien Tesnière

Source: Wikimedia Commons [1] [2]

Information extraction

- **Information extraction (IE)** is the task of extracting structured information from running text.

- More specifically, the term ‘structured information’ refers to **named entities** and **semantic relations** between those entities.

  persons, organisations, companies – *X* is-leader-of *Y*, *X* bought *Y*

JEOPARDY!

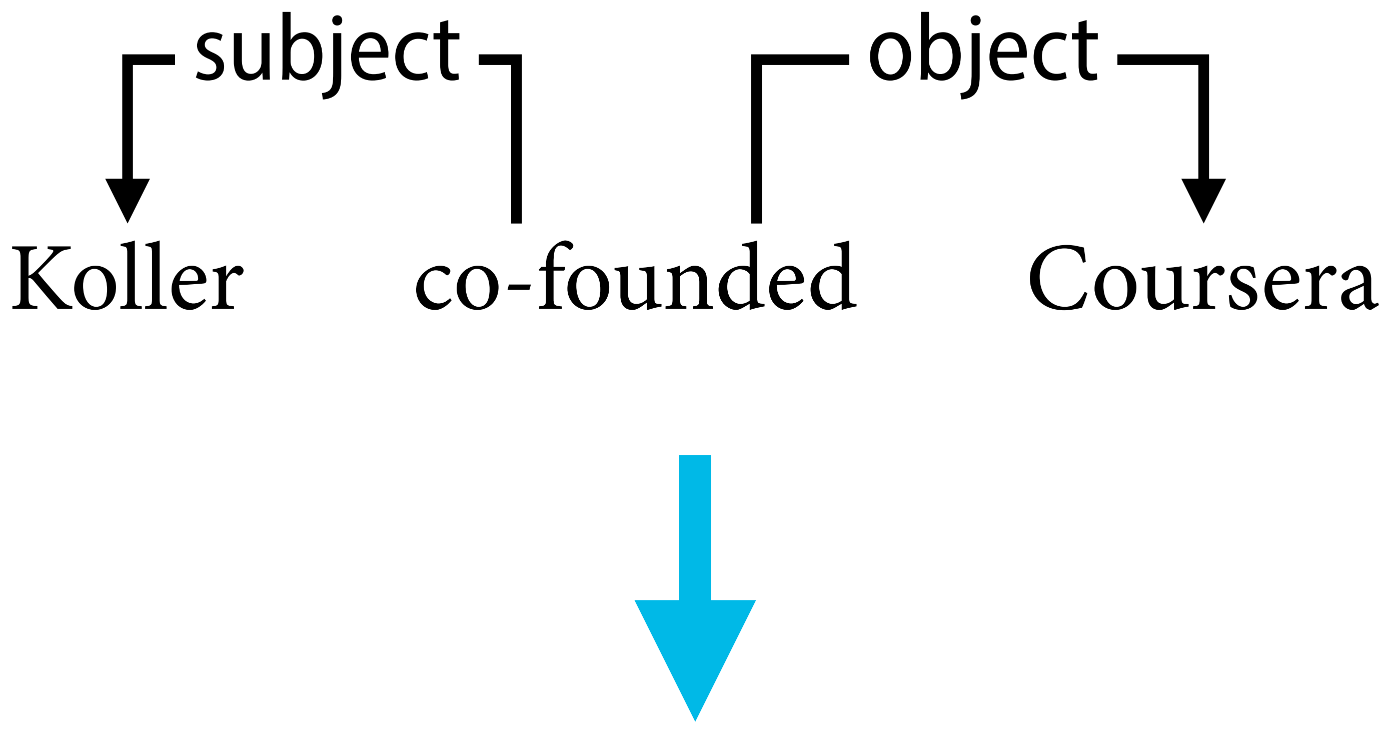
This Stanford University alumnus co-founded educational technology company Coursera.

Source: MacArthur Foundation

SPARQL query against DBPedia

```
SELECT DISTINCT ?x WHERE {
  ?x dbo:almaMater dbr:Stanford_University.
  dbr:Coursera dbo:foundedBy ?x.
}
```

Syntactic structure, semantic relations



```
dbr:Coursera dbo:foundedBy dbr:Daphne_Koller
```

Algorithmic approaches to syntactic analysis

- **Exhaustive search**

  Cast parsing as a combinatorial optimisation problem over the set of target representations (trees).

  CKY algorithm

- **Greedy search**

  Casts parsing as a sequence of classification problems: at each point in time, predict one of several parser actions.

  transition-based dependency parsing

Google Research Blog

The latest news from Research at Google

Announcing SyntaxNet: The World's Most Accurate Parser Goes Open Source

Thursday, May 12, 2016

Posted by Slav Petrov, Senior Staff Research Scientist

At Google, we spend a lot of time thinking about how computer systems can read and understand human language in order to process it in intelligent ways. Today, we are excited to share the fruits of our research with the broader community by releasing SyntaxNet, an open-source neural network framework implemented in TensorFlow that provides a foundation for Natural Language Understanding (NLU) systems. Our release includes all the code needed to train new SyntaxNet models on your own data, as well as *Parsey McParseface*, an English parser that we have trained for you and that you can use to analyze English text.

This lecture

- Introduction to syntactic analysis
- Parsing to phrase structure trees
    - Context-free grammars
    - Parsing with probabilistic context-free grammars
- Parsing to dependency trees
    - Transition-based dependency parsing

Context-free grammars

Phrases and syntactic heads

- Words within sentences form groupings called **phrases**.

  Kim read [a book]. Kim read [a very interesting book about grammar].

- Each phrase is projected by a **syntactic head**, which determines its internal structure and external distribution.

  [The <u>war</u> on drugs] is controversial. / *[The <u>battle</u> on drugs] is controversial.

  [The <u>war</u> on drugs] is controversial. / *[The <u>war</u> on drugs] are controversial.

Context-free grammars

- Phrases can be combined to form larger phrases. This gives rise to a hierarchical structure.
- The phrase structure of a sentence can be described using **context-free grammars**.
- The main ingredient of a context-free grammar is a set of rules that describe how phrases are structured.

A context-free grammar

Rule	Example
S → NP VP	I + want a morning flight
NP → Pronoun	I
NP → Proper-Noun	Los Angeles
NP → Det Nominal	a flight
Nominal → Nominal Noun	morning flight
Nominal → Noun	flights
VP → Verb	do
VP → Verb NP	want + a flight
VP → Verb NP PP	leave + Boston + in the morning
VP → Verb PP	leaving + on Thursday
PP → Preposition NP	from + Los Angeles

Context-free grammars, formal definition

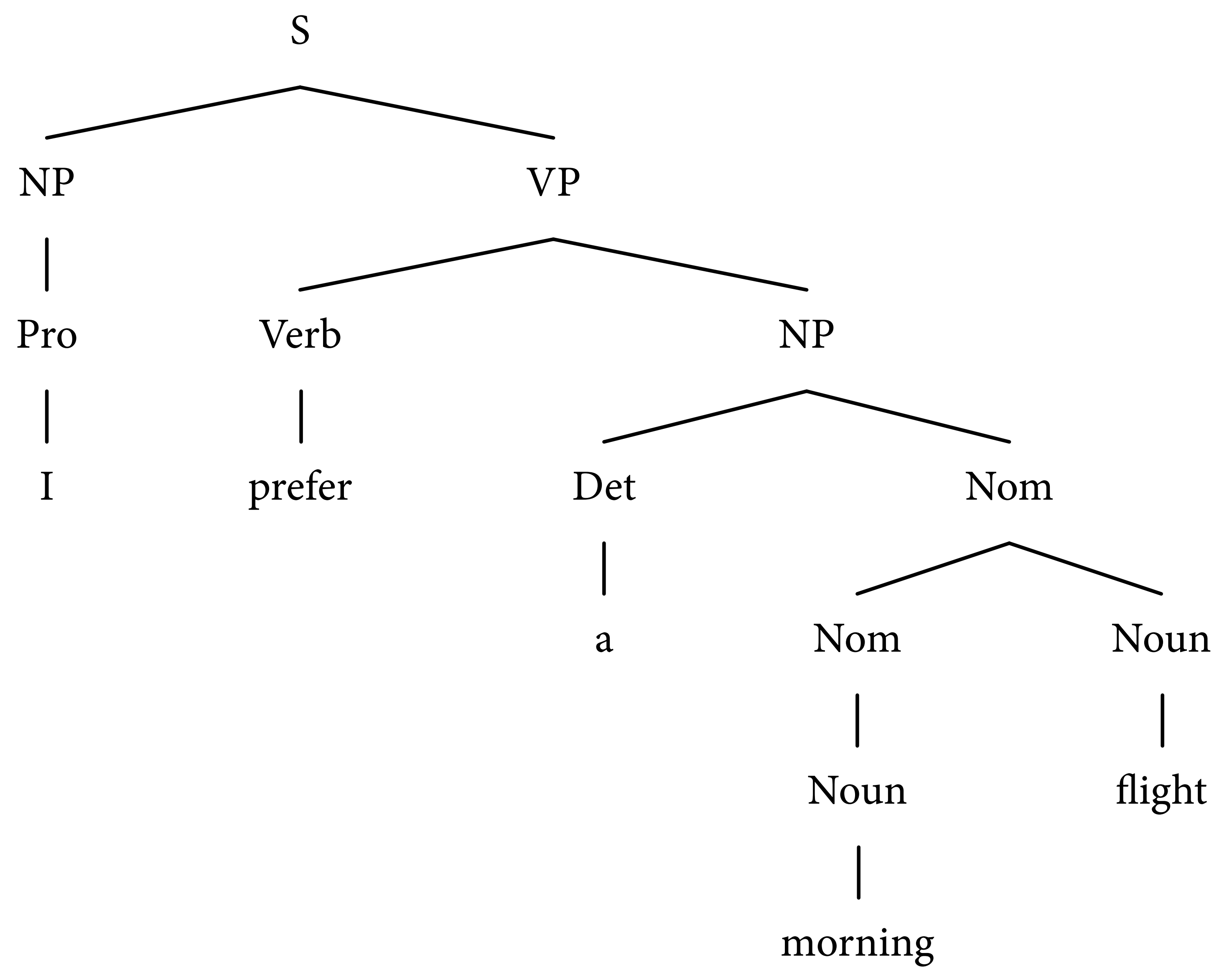
N	a set of nonterminals (phrase labels)
T	a set of terminals (words)
P	a finite set of rules or productions
S	a distinguished nonterminal symbol called the start symbol

Notation for rules

$$\overset{\text{left-hand side}}{\overset{\downarrow}{\mathrm{S}}} \rightarrow \mathrm{NP}\ \underset{\text{right-hand side}}{\underset{\uparrow}{\mathrm{VP}}}$$

‘A sentence (S) consists of a noun phrase (NP) and a verb phrase (VP).’

Phrase structure tree



Limitations of context-free grammars

- Context-free grammars can model many important aspects of natural language syntax.

  linguistic creativity, nested structures

- But there are other aspects that they do not model adequately, or are unable to model at all.

  agreement, crossing dependencies

Subject–verb agreement

- In English, a verb and its grammatical subject need to agree with respect to number.

  *[A flight] [leave Boston in the morning]

- The rules of our example grammar do not capture this regularity. The grammar **overgenerates**.

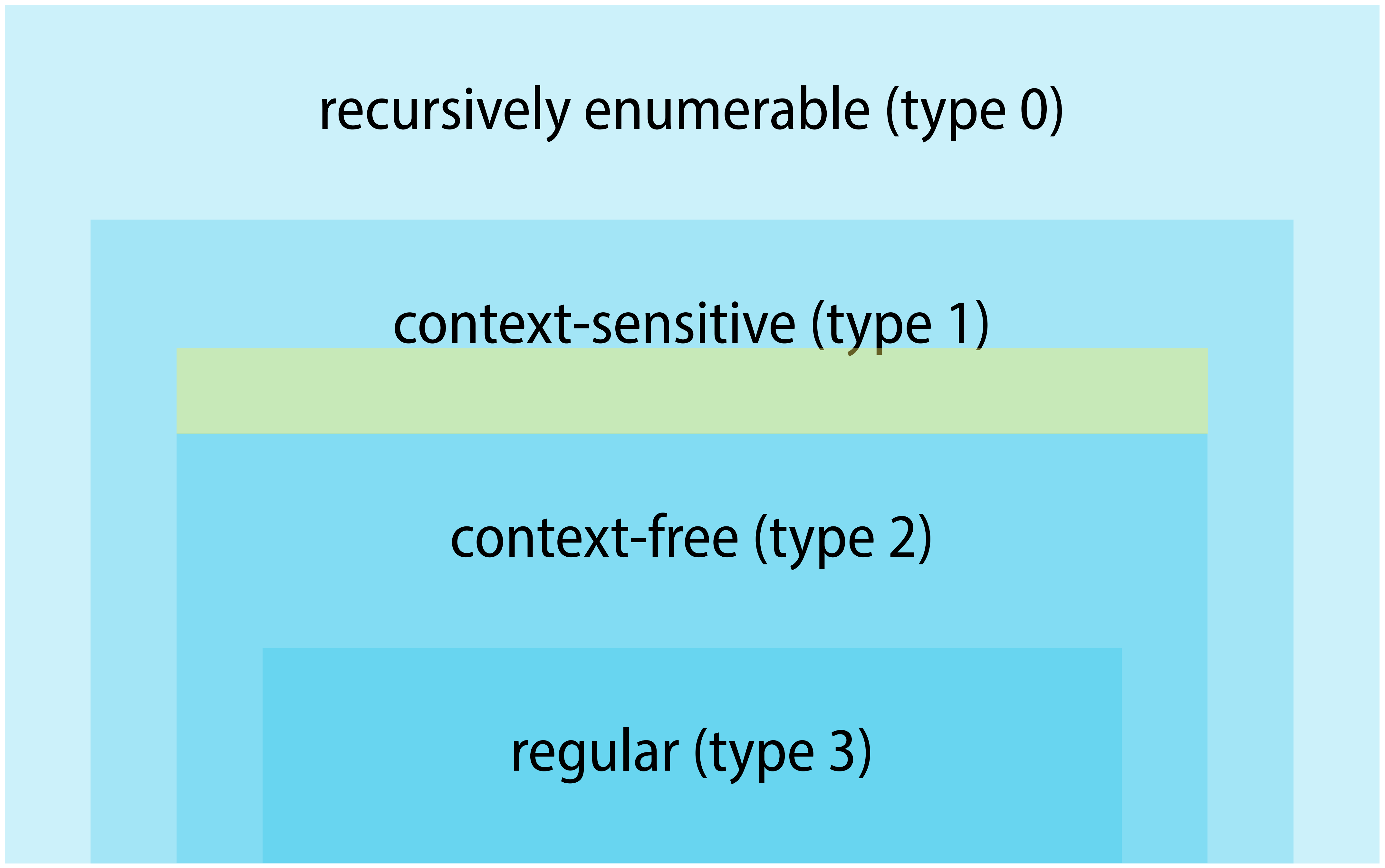
Subject–verb agreement

One way to solve the problem with overgeneration is to specialise the rules of the grammar with respect to number:

Rule	Example
S → NP[sg] VP[sg]	this flight + leaves on Monday
NP[sg] → Det[sg] Nom[sg]	this + flight
VP[sg] → Verb[sg] PP	leaves + on Monday
NP[pl] → Det[pl] NP[pl]	these + flights

However, this makes the size of grammar explode.

Chomsky hierarchy

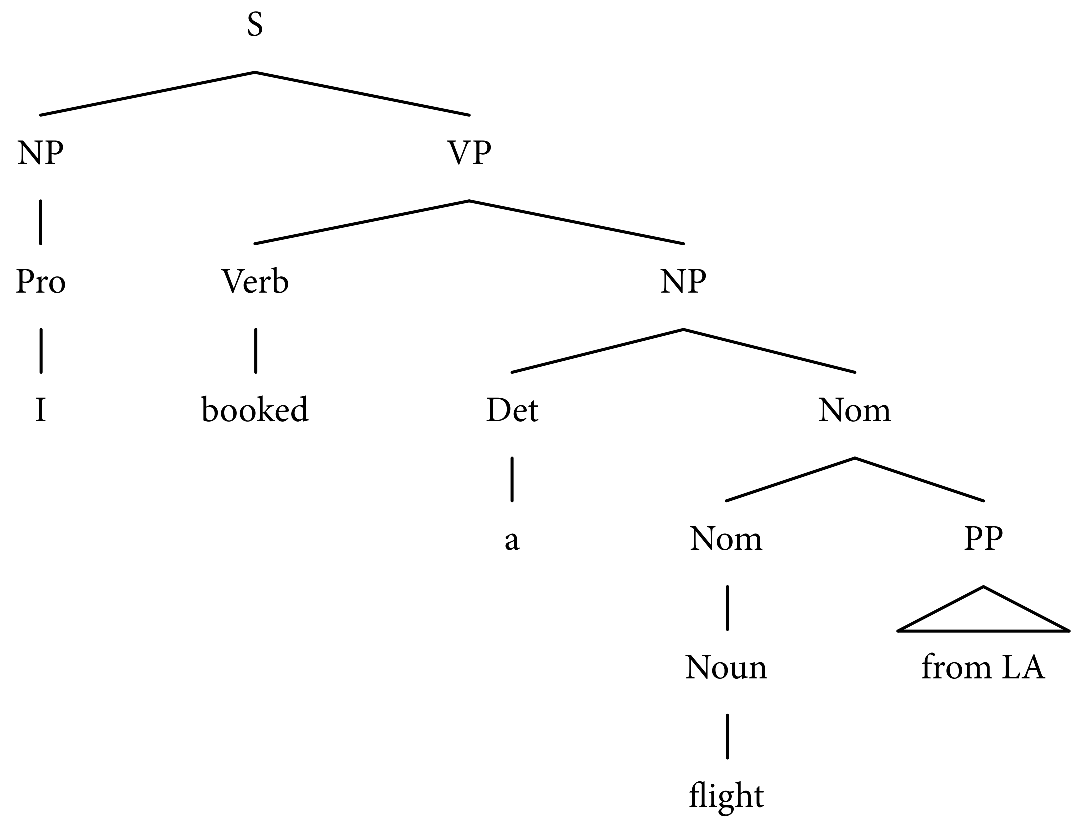


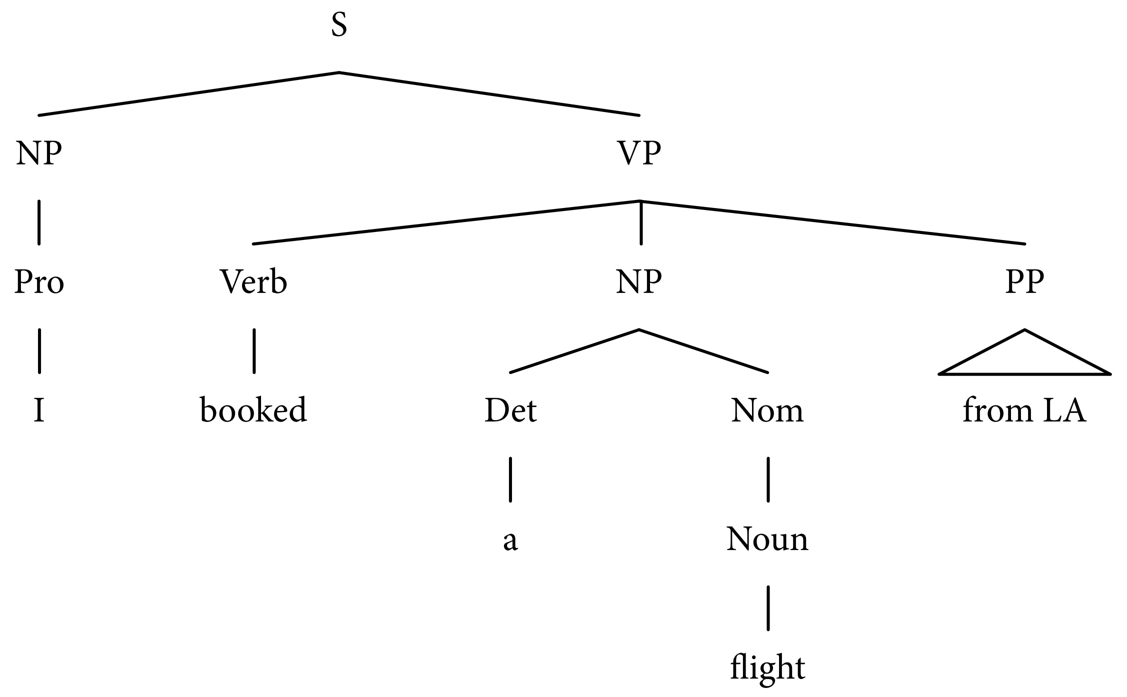
This lecture

- Introduction to syntactic analysis
- Parsing to phrase structure trees
  - Context-free grammars
  - Parsing with probabilistic context-free grammars
- Parsing to dependency trees
  - Transition-based dependency parsing

Parsing with probabilistic context-free grammars

Syntactic ambiguity

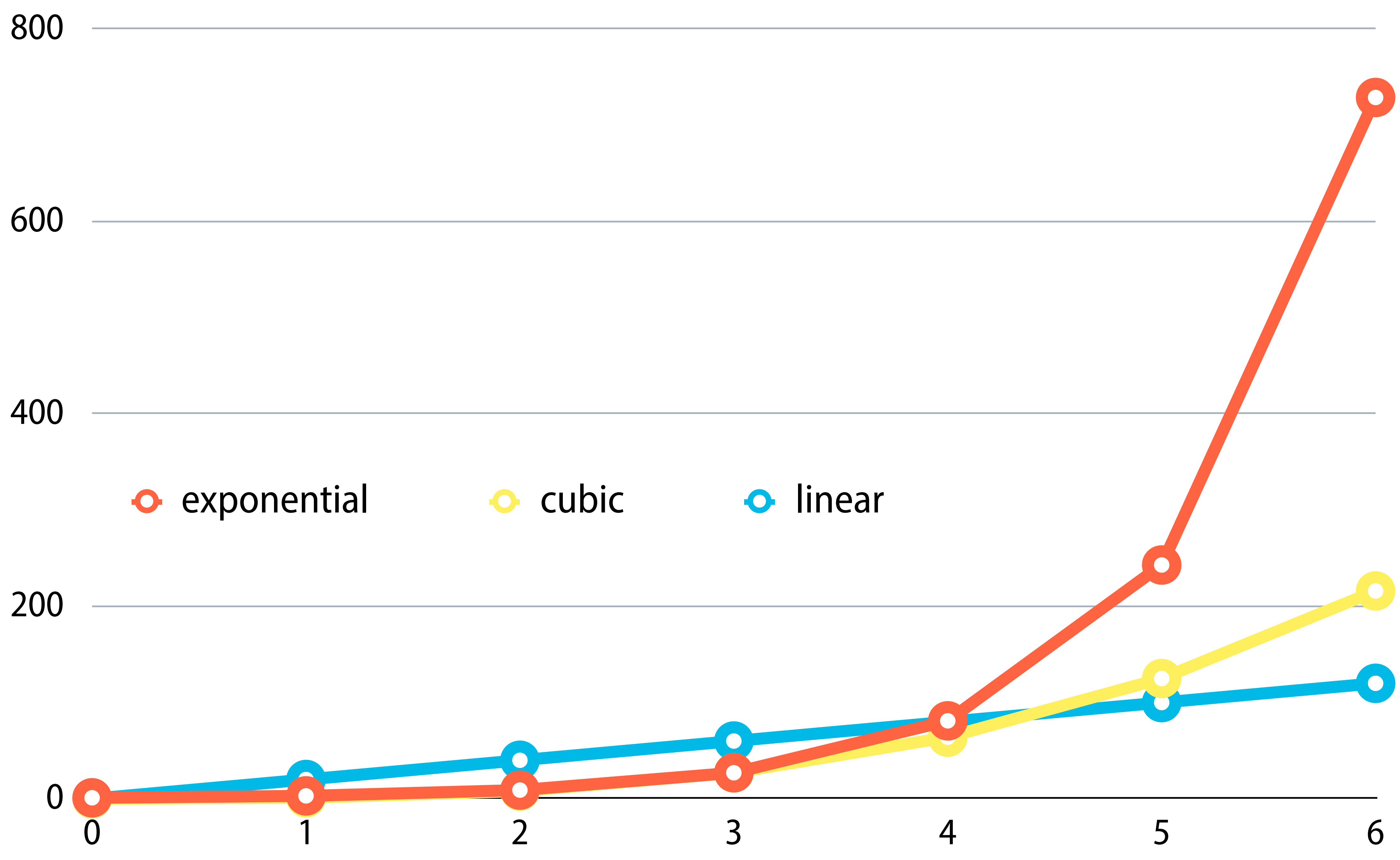




The PP modifies ‘flight’.

The PP modifies ‘booked’.

Combinatorial explosion



Probabilistic grammars

- The number of possible parse trees grows exponentially with the length of the sentence.
- But not all parse trees are equally relevant, and in many applications, we just want to find the most probable parse tree.

Probabilistic context-free grammar

A **probabilistic context-free grammar (PCFG)** is a context-free grammar with the following additional properties:

- Every rule r has been assigned a probability $P(r)$.
- The total probability of all rules with the same left-hand side is 1.

Probabilistic context-free grammar

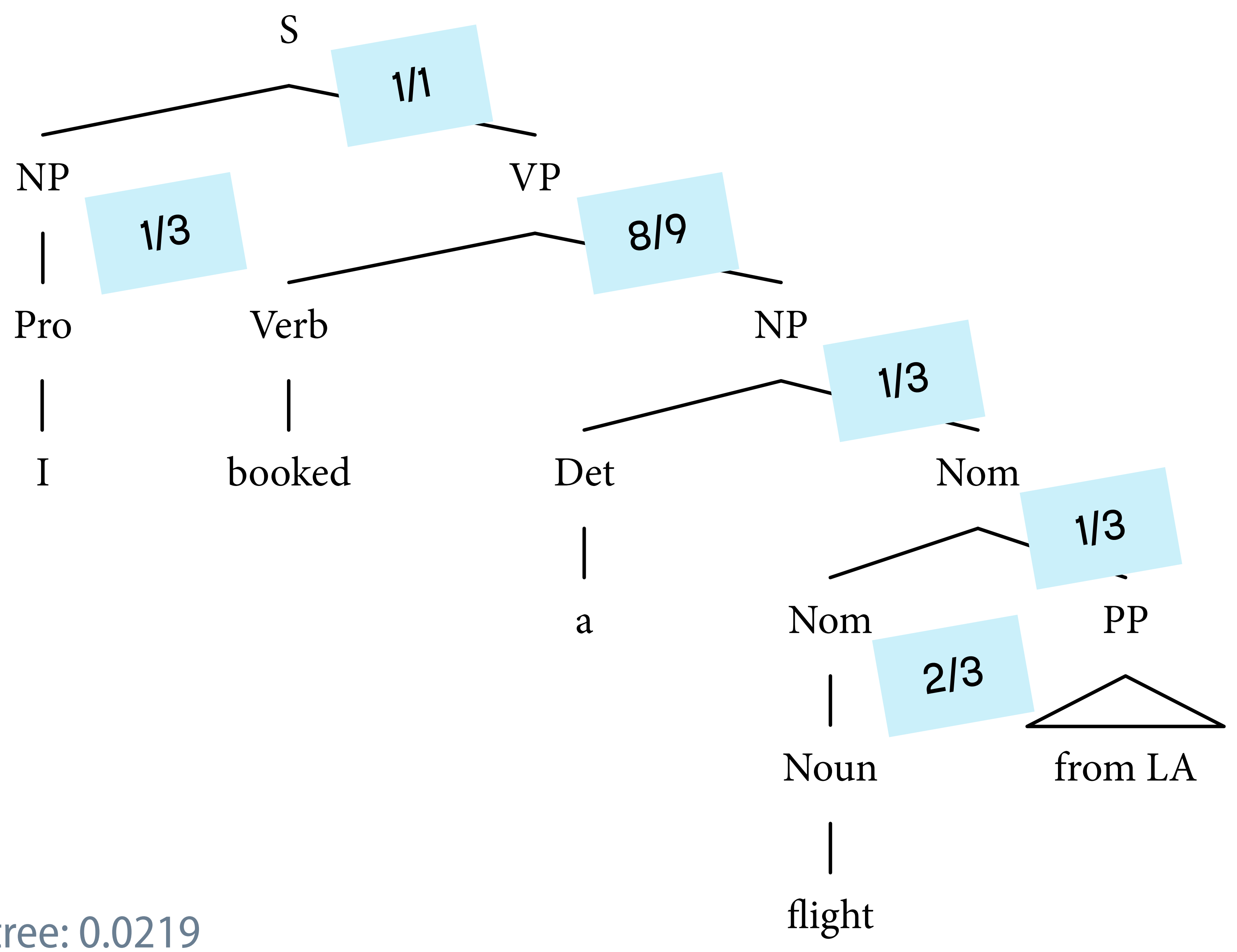
Rule	Probability
S → NP VP	1/1
NP → Pronoun	1/3
NP → Proper-Noun	1/3
NP → Det Nominal	1/3
Nominal → Nominal PP	1/3
Nominal → Noun	2/3
VP → Verb NP	8/9
VP → Verb NP PP	1/9
PP → Preposition NP	1/1

The probability of a parse tree

The probability of a parse tree t is defined as the product of the probabilities of the rules that appear in t:

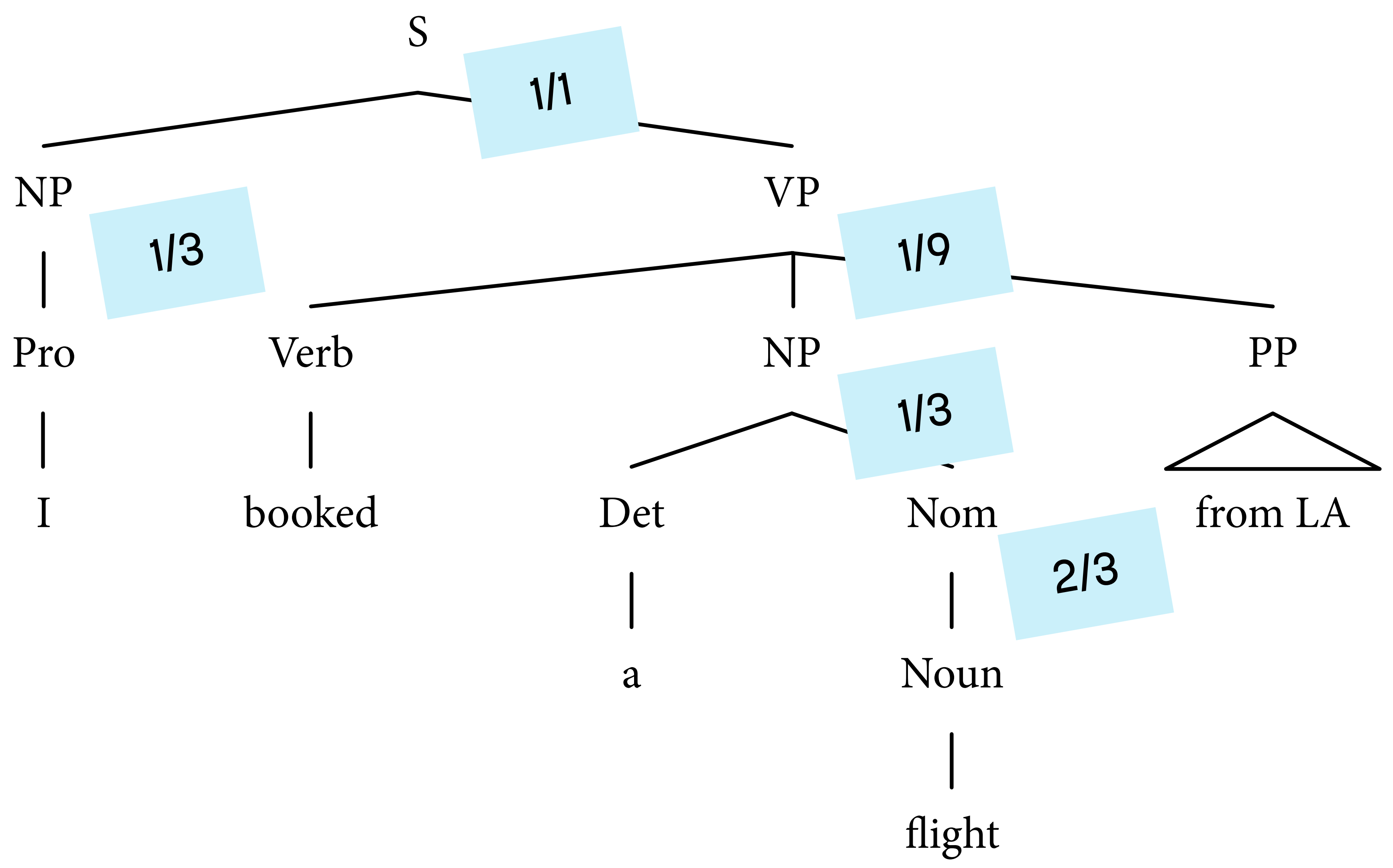
$$P(t) = \prod_{r \in t} P(r)$$

Probability of a parse tree



Probability of this tree: 0.0219

Probability of a parse tree



Probability of this tree: 0.0082

The CKY algorithm

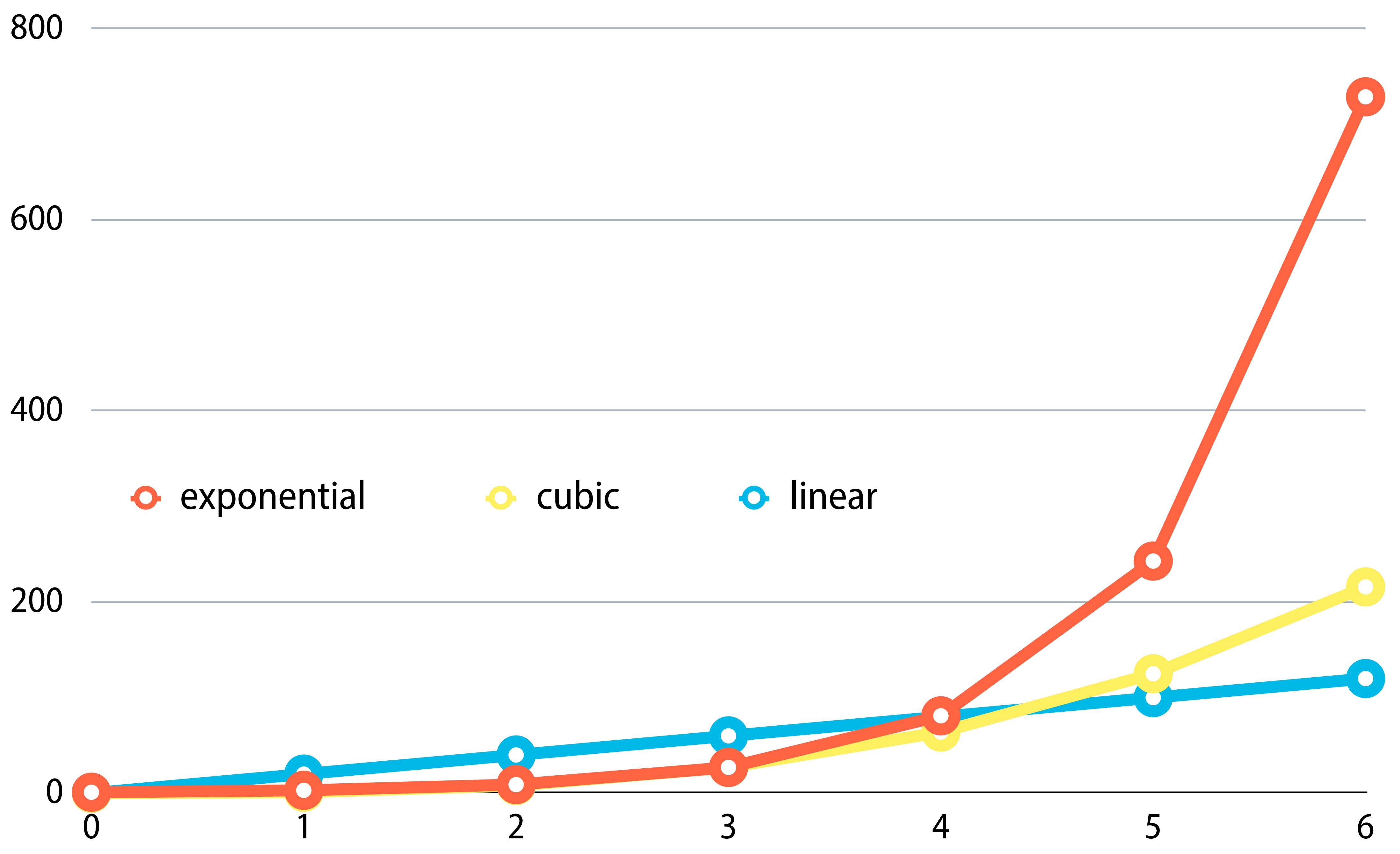
- We need an efficient algorithm that can find the most probable parse tree, much like the Viterbi algorithm for POS tagging.

  efficient = runtime is at most polynomial in the length of the sentence

- One such algorithm is (the probabilistic extension of) the **Cocke–Kasami–Younger (CKY) algorithm**.

  advanced material

Combinatorial explosion



Treebanks

- Until the mid-1990s, syntactic parsers used large, hand-written grammars created by linguistic experts.
- Modern parsers are learned from corpora of syntactic analyses called **treebanks**.

  Penn Treebank, Swedish Treebank, Universal Dependencies Project

Penn Treebank

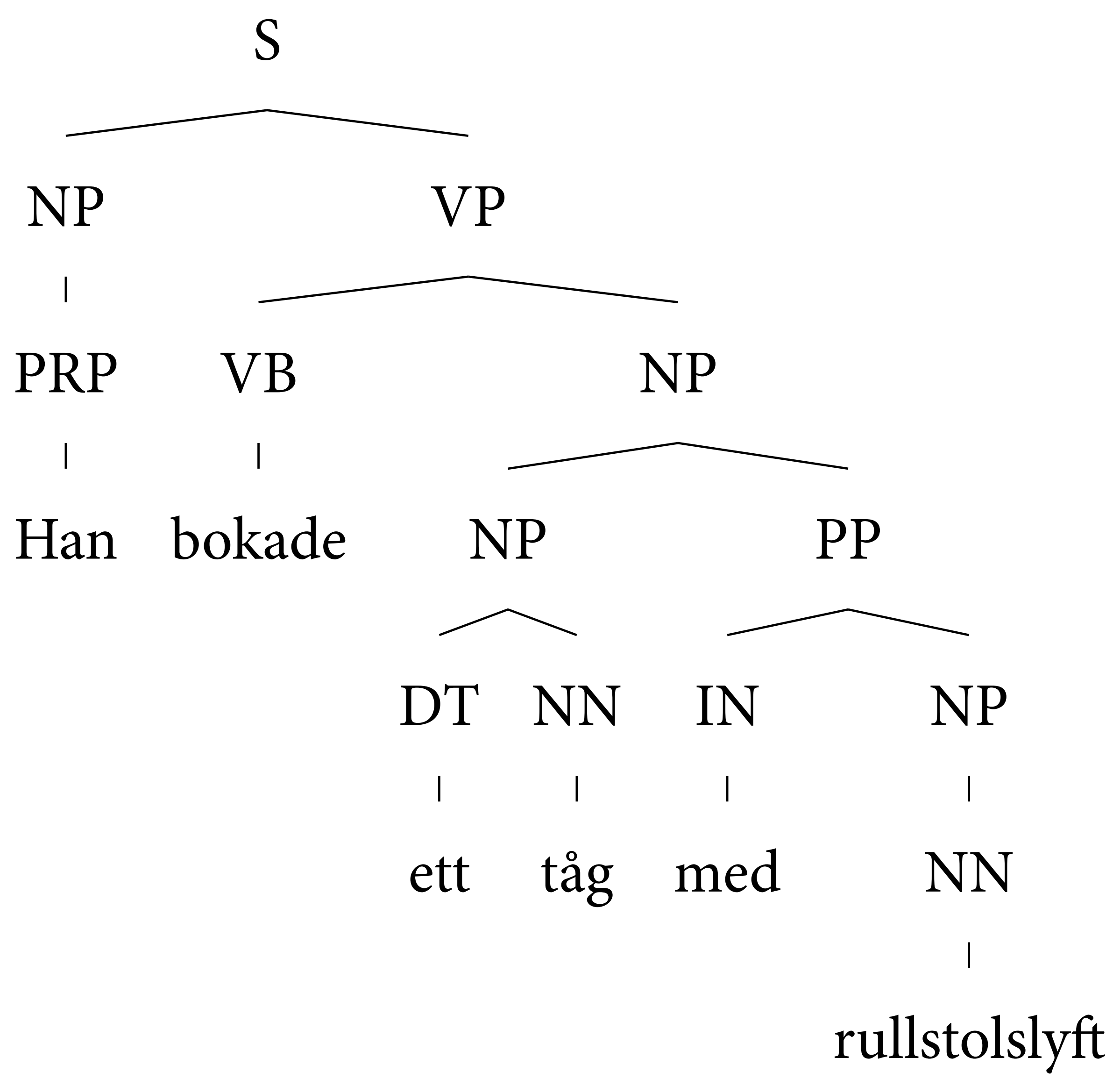
```
( (S
    (NP-SBJ
      (NP (NNP Pierre) (NNP Vinken) )
      (, ,)
      (ADJP
        (NP (CD 61) (NNS years) )
        (JJ old) )
      (, ,) )
    (VP (MD will)
      (VP (VB join)
        (NP (DT the) (NN board) )
        (PP-CLR (IN as)
          (NP (DT a) (JJ nonexecutive) (NN director) ))
        (NP-TMP (NNP Nov.) (CD 29) )))
    (. .) ))
```

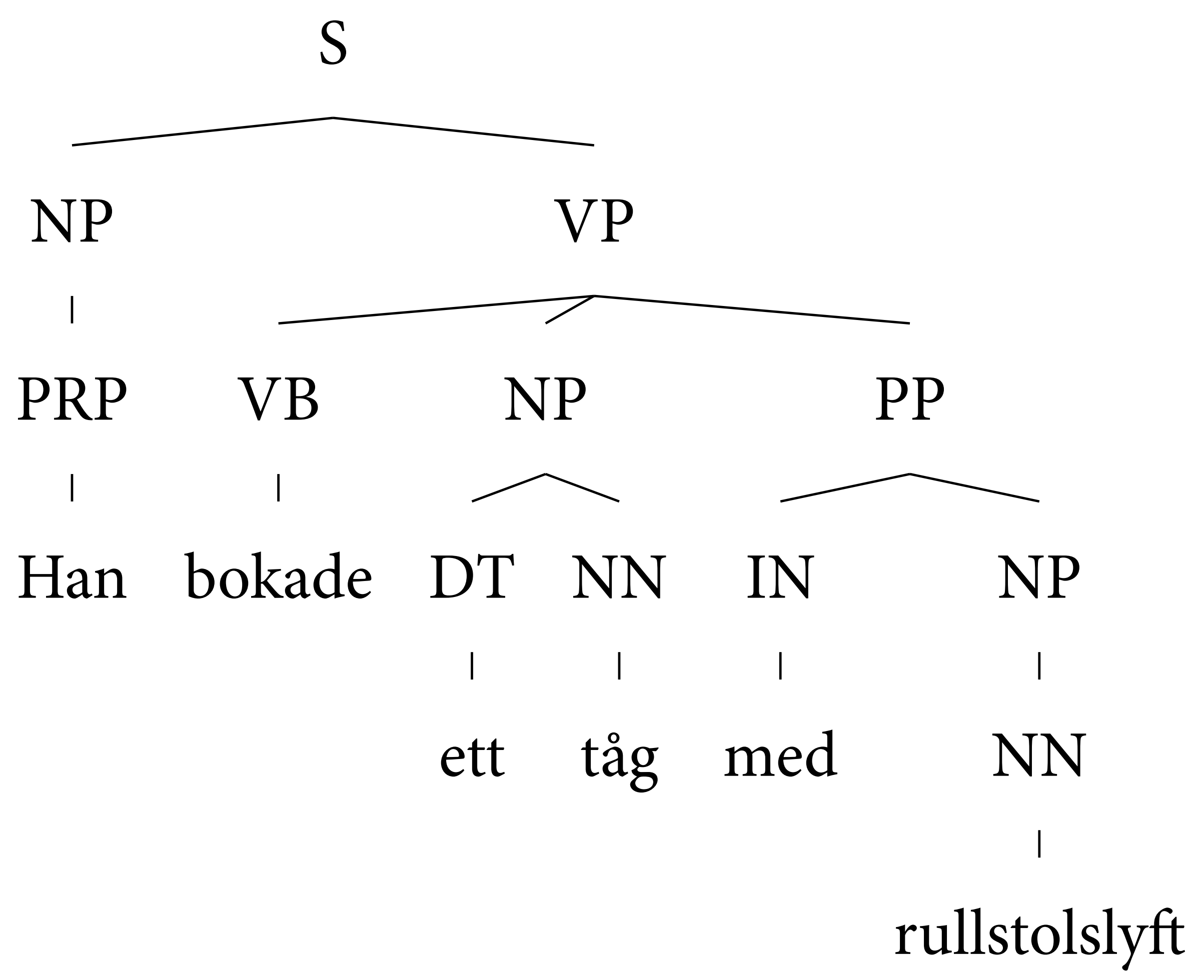
Grammar rule	Phrase
S → NP-SBJ VP .	Pierre Vinken … Nov. 29.
NP-SBJ → NP , ADJP ,	Pierre Vinken, 61 years old,
VP → MD VP	will join the board …
NP → DT NN	the board

Estimation of rule probabilities

- Given a phrase structure treebank, the rule probabilities of a PCFG can be obtained using maximum likelihood estimation.
- To do this, we divide the count for a certain rule by the count for all rules that share the same left-hand side.

Sample exam question: Estimate rule probabilities





This lecture

- Introduction to syntactic analysis
- Parsing to phrase structure trees
  - Context-free grammars
  - Parsing with probabilistic context-free grammars
- Parsing to dependency trees
  - Transition-based dependency parsing

Transition-based dependency parsing

Algorithmic approaches

- **Exhaustive search**

  Cast parsing as a combinatorial optimisation problem over the set of target representations (trees).

  CKY algorithm

- **Greedy search**

  Cast parsing as a sequence of classification problems: at each point in time, predict one of several parser actions.

  transition-based dependency parsing

Dependency parsing as classification

- In Section 3 we have seen how part-of-speech tagging can be broken down into a sequence of classification problems.

  part-of-speech tagging with the multi-class perceptron

- In this section we will see how the same idea can be applied to dependency parsing.

- Instead of POS tags, the classifier will predict **transitions** that take the parser from one **configuration** to another.

  moves, states

Google Research Blog

The latest news from Research at Google

Announcing SyntaxNet: The World's Most Accurate Parser Goes Open Source

Thursday, May 12, 2016

Posted by Slav Petrov, Senior Staff Research Scientist

At Google, we spend a lot of time thinking about how computer systems can read and understand human language in order to process it in intelligent ways. Today, we are excited to share the fruits of our research with the broader community by releasing SyntaxNet, an open-source neural network framework implemented in TensorFlow that provides a foundation for Natural Language Understanding (NLU) systems. Our release includes all the code needed to train new SyntaxNet models on your own data, as well as *Parsey McParseface*, an English parser that we have trained for you and that you can use to analyze English text.

Transition-based dependency parsing

- The parser starts in the **initial configuration**.
- It then calls the classifier, which predicts the transition that the parser should make to move to the next configuration.
- This process is repeated until the parser reaches a **terminal configuration**.

Configurations

A parser configuration consists of three parts:

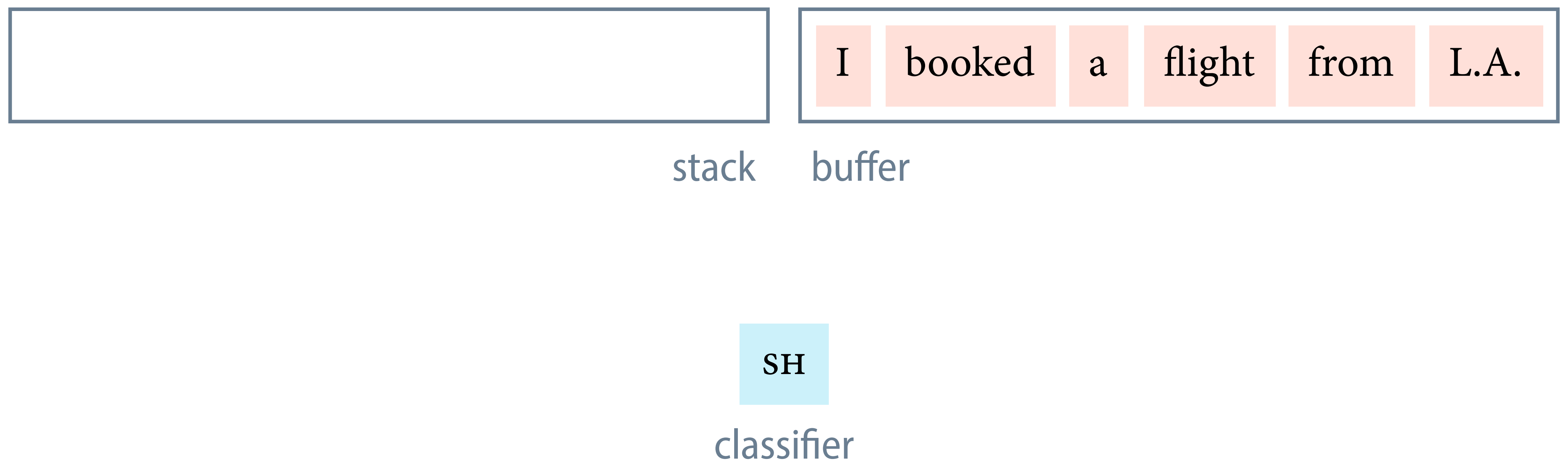
- A **buffer**, which contains those words in the sentence that still need to be processed. Initially, the buffer contains all words.
- A **stack**, which contains those words in the sentence that are currently being processed. Initially, the stack is empty.
- A **partial dependency tree**. Initially, this tree contains all the words of the sentence, but no dependency arcs.

Transitions

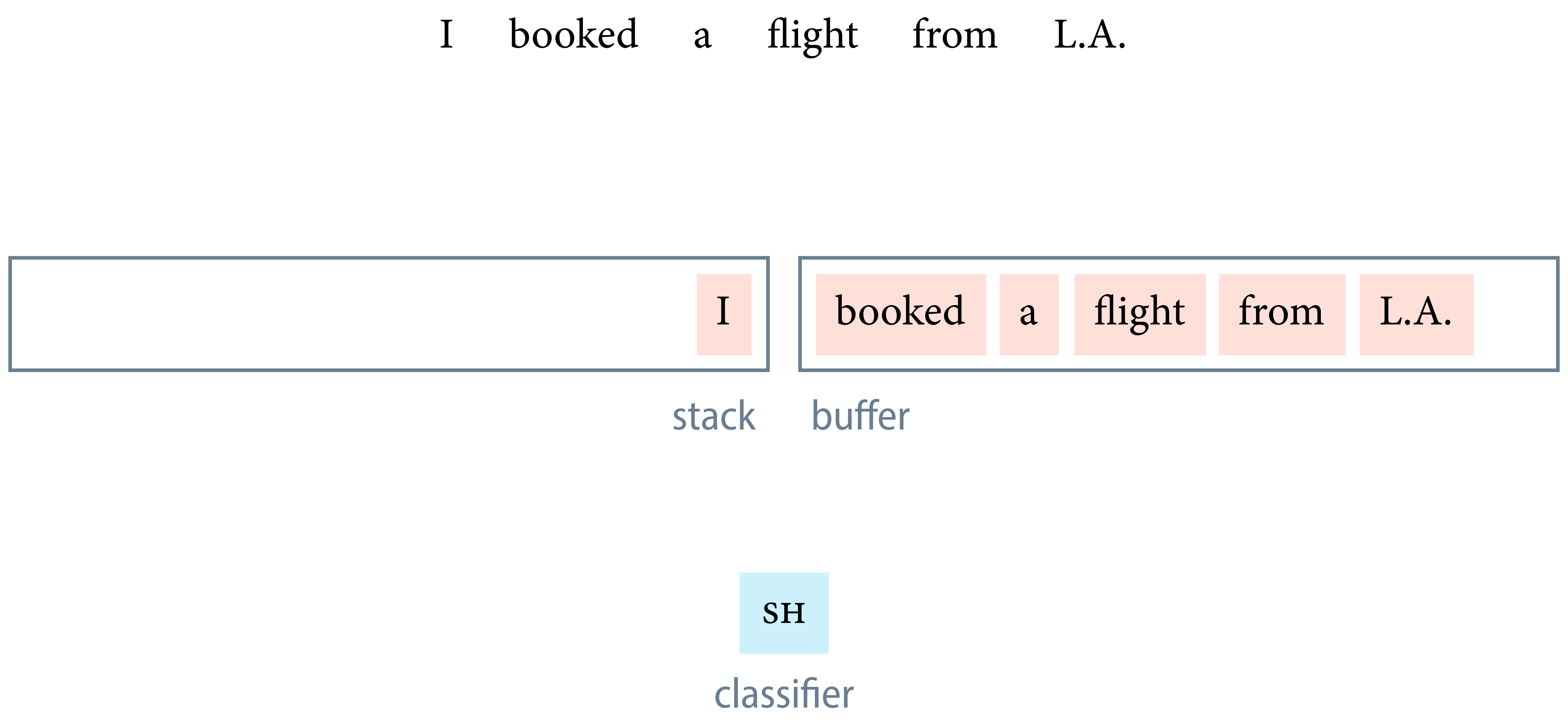
- The **shift transition (SH)** removes the frontmost word from the buffer and pushes it to the top of the stack.
- The **left-arc transition (LA)** creates a dependency from the topmost word on the stack to the second-topmost word, and removes the second-topmost word.
- The **right-arc transition (RA)** creates a dependency from the second-topmost word on the stack to the topmost word, and removes the topmost word.

Transition-based dependency parsing, example

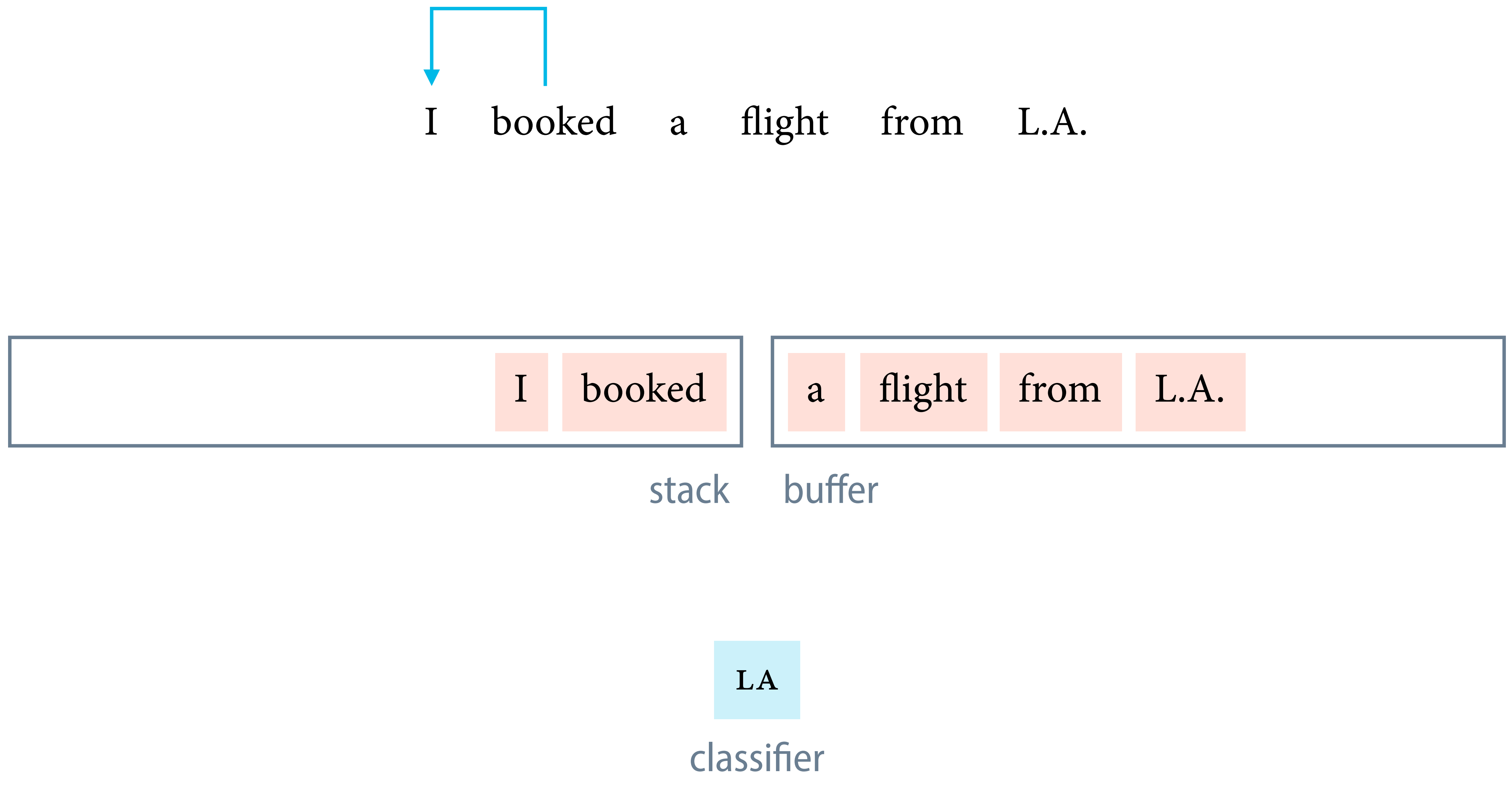
I booked a flight from L.A.



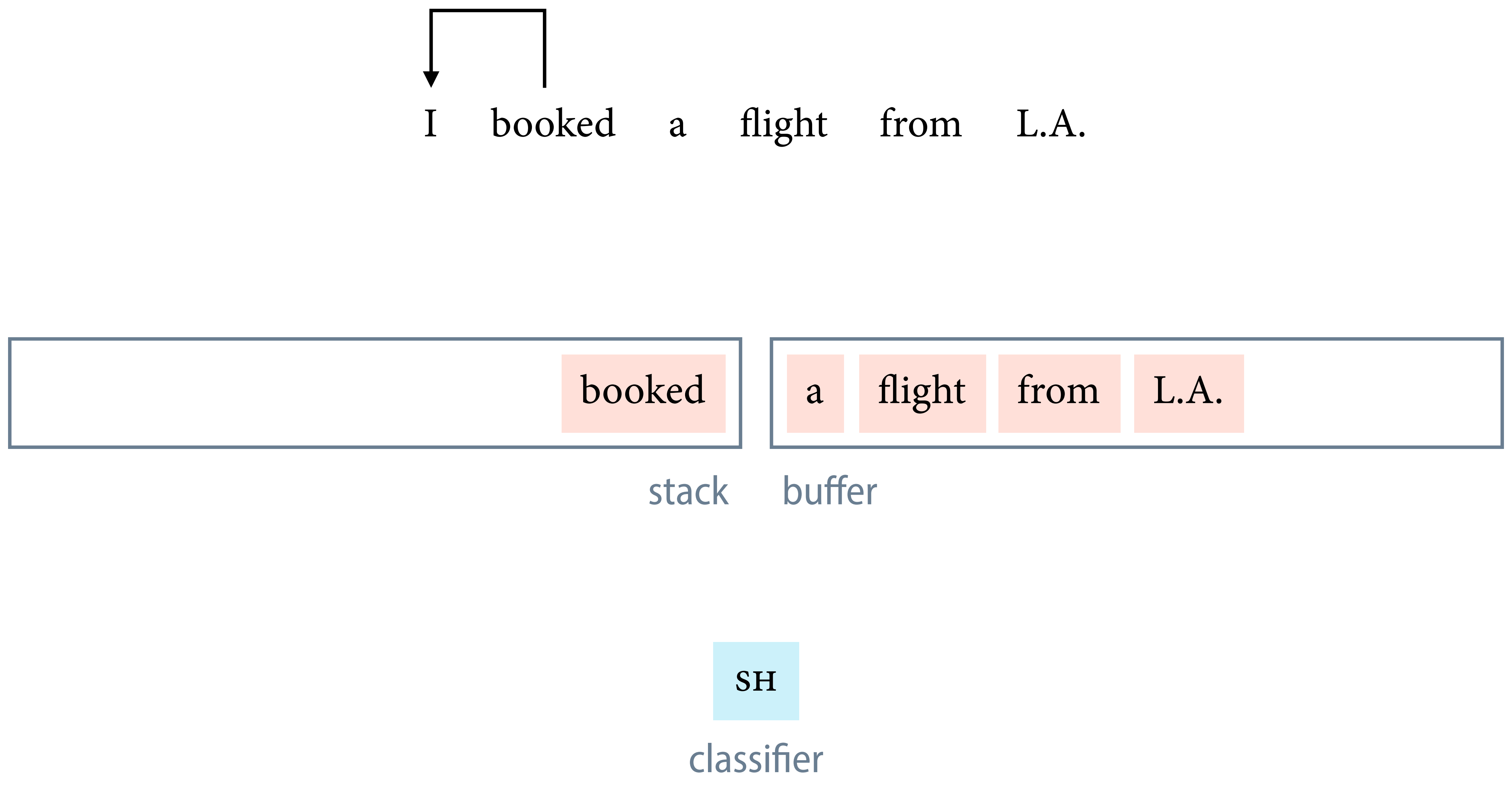
Transition-based dependency parsing, example



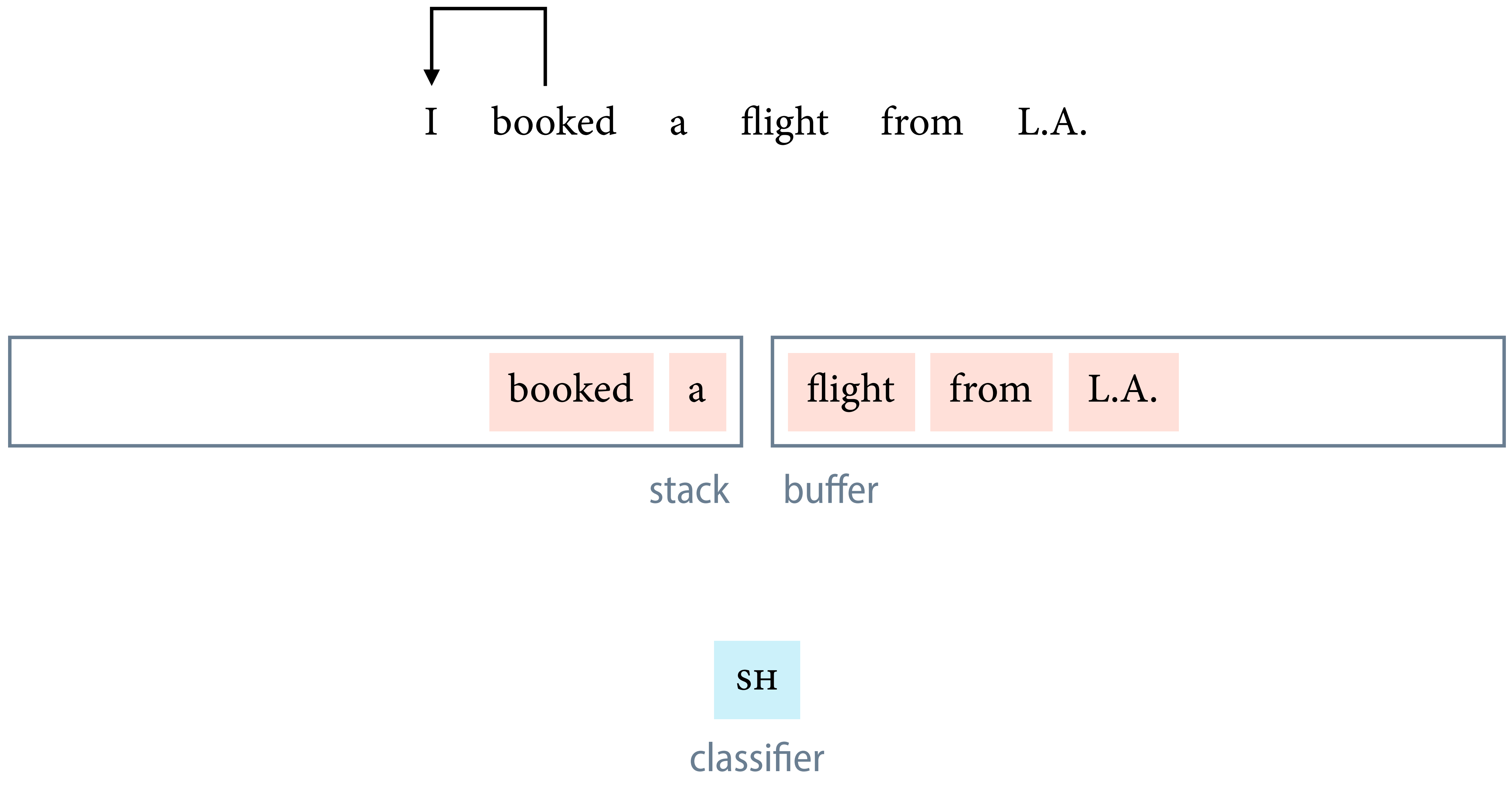
Transition-based dependency parsing, example



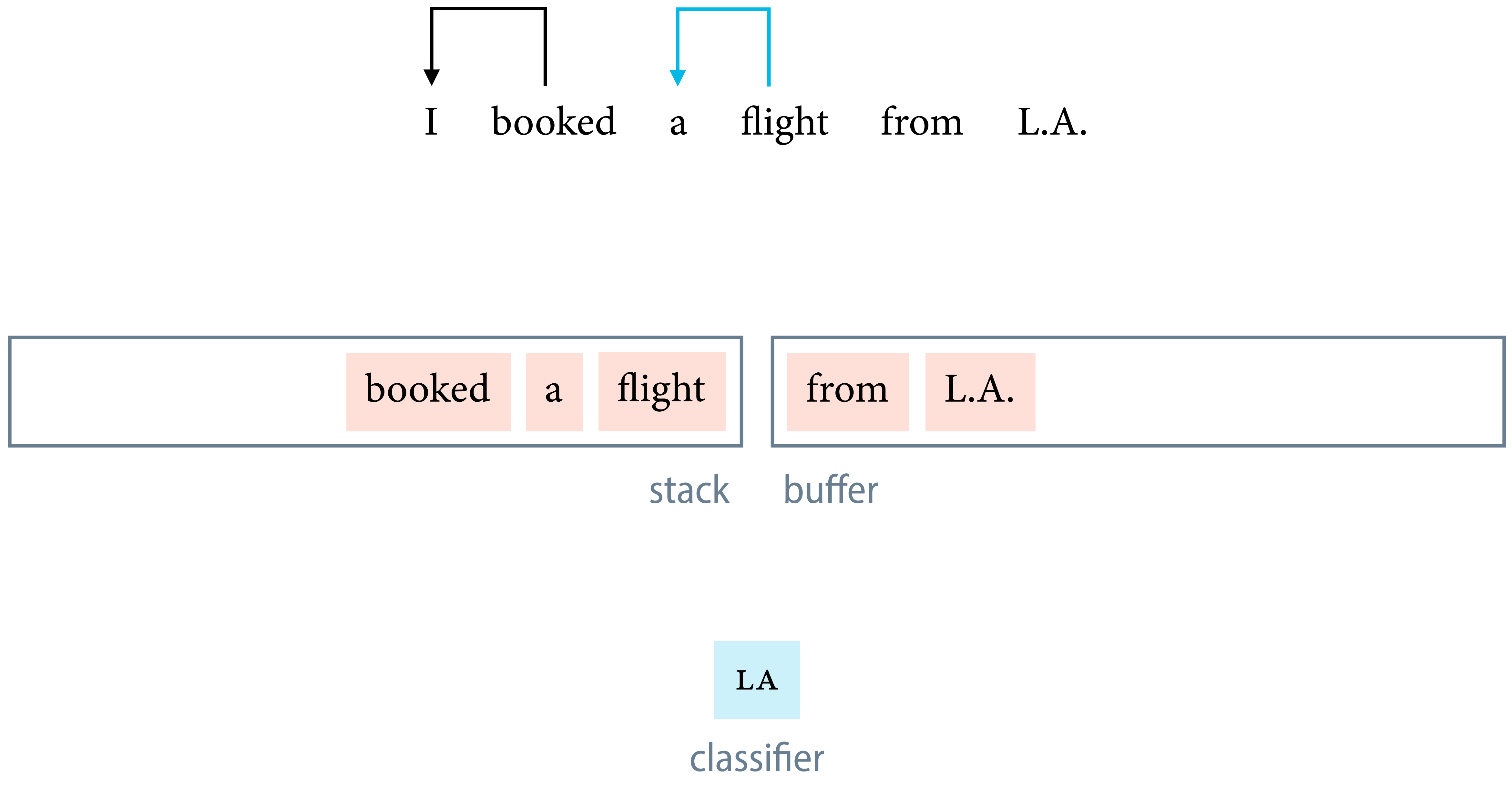
Transition-based dependency parsing, example



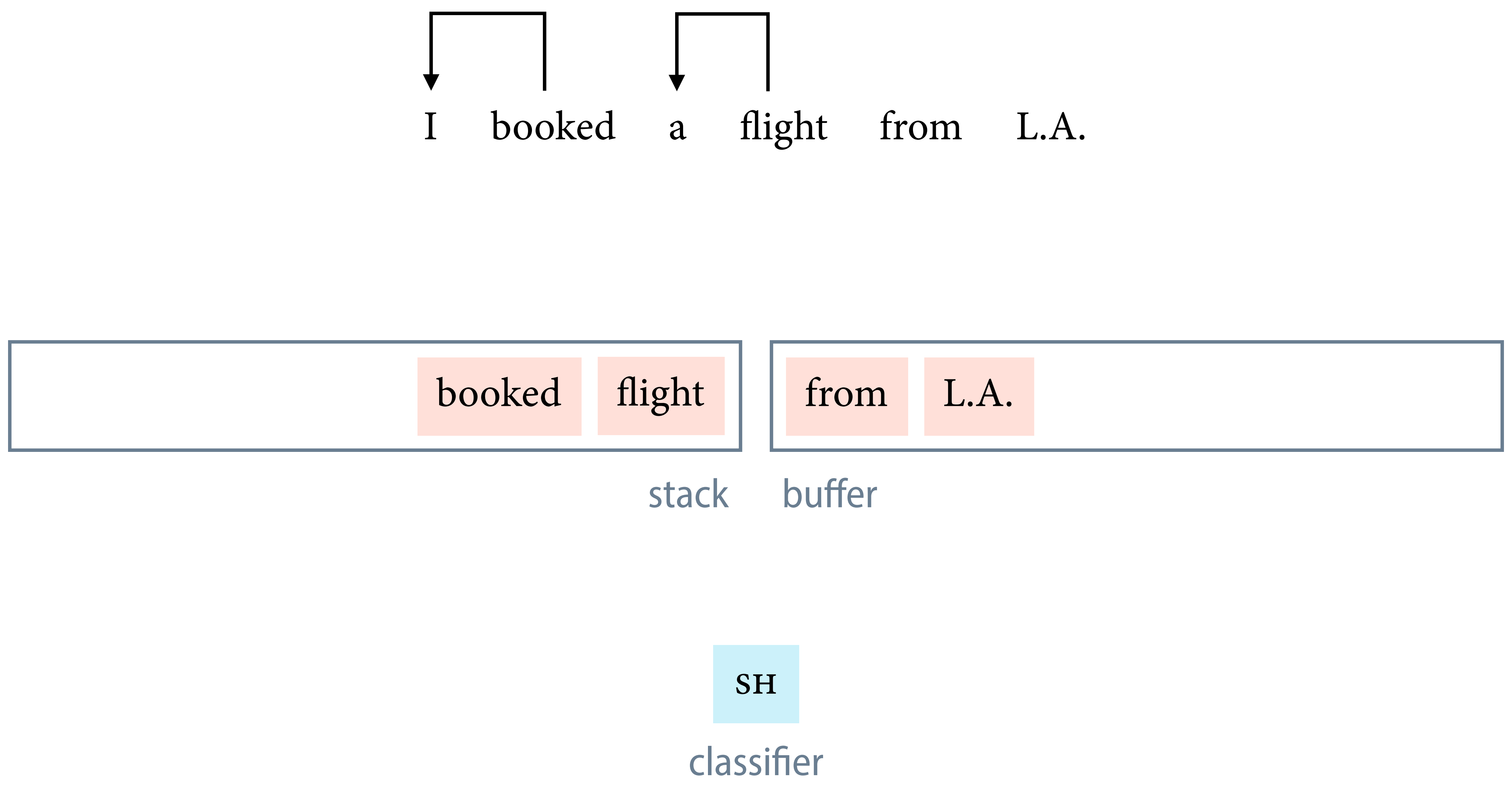
Transition-based dependency parsing, example



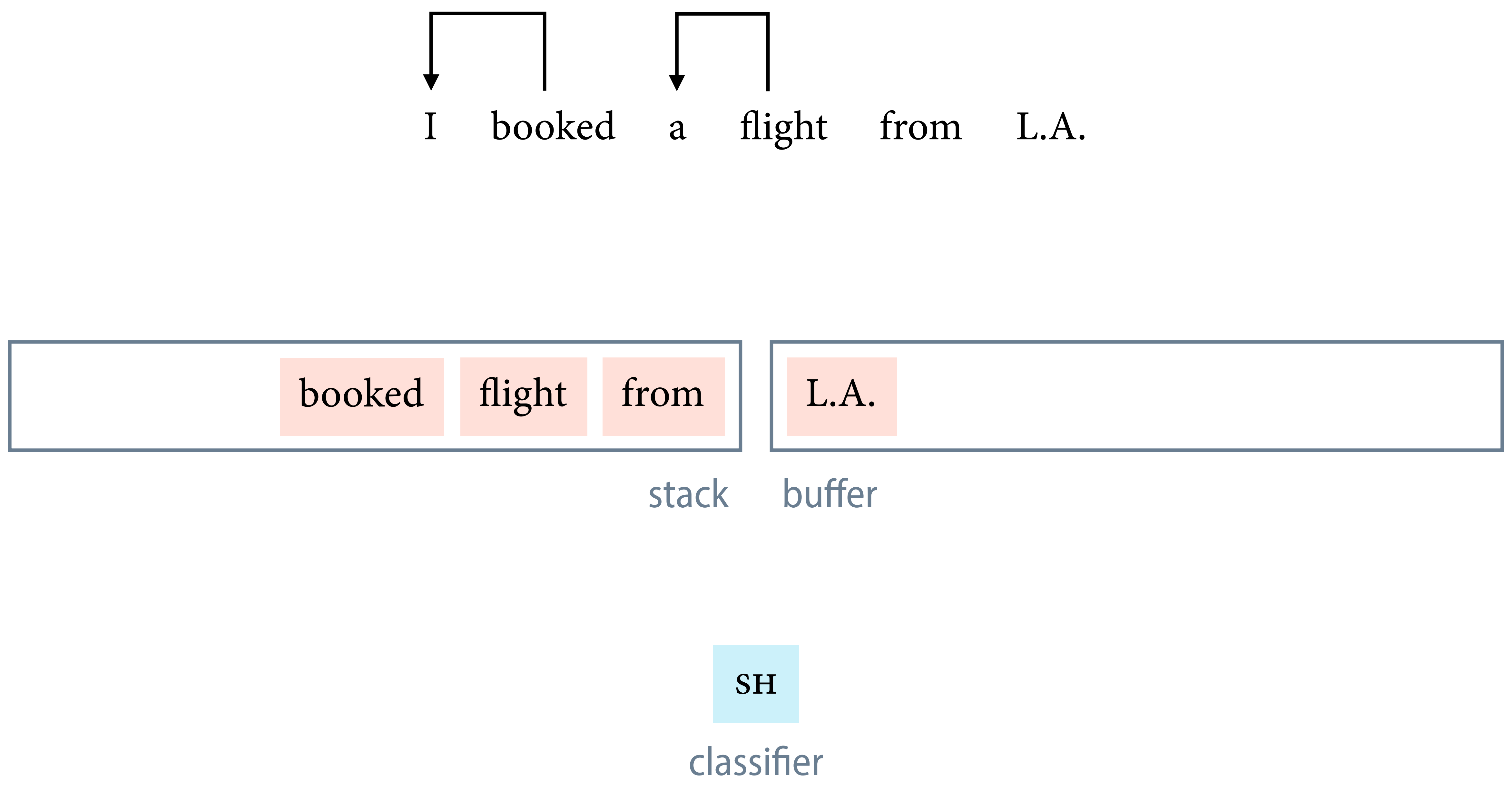
Transition-based dependency parsing, example



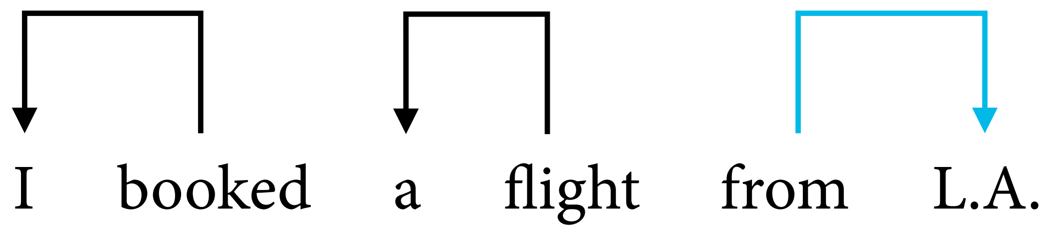
Transition-based dependency parsing, example

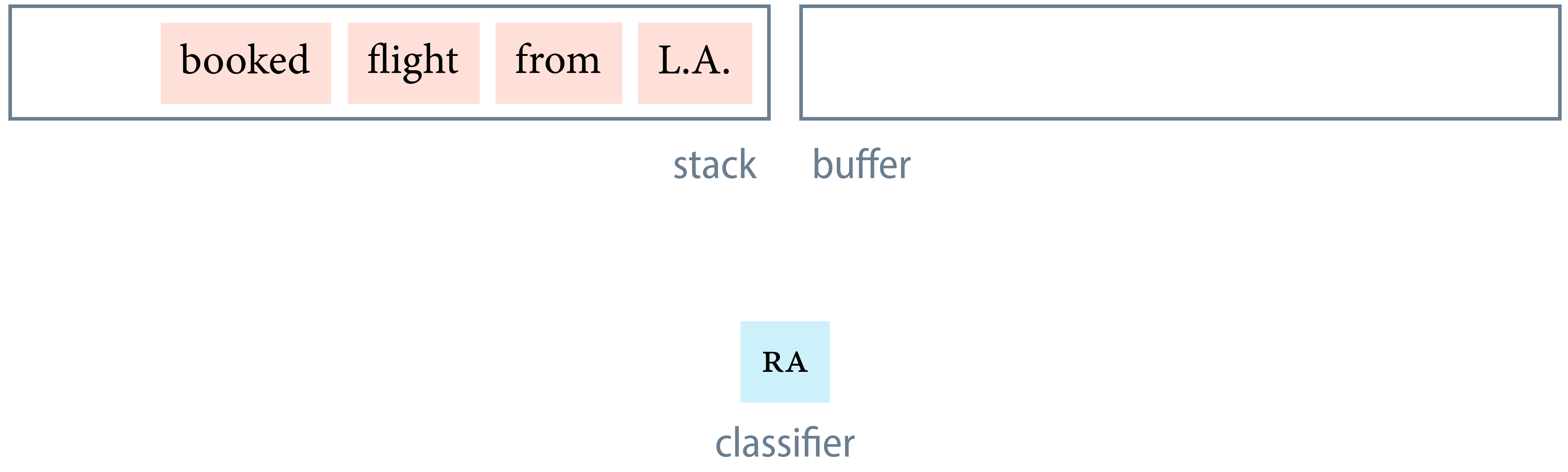


Transition-based dependency parsing, example

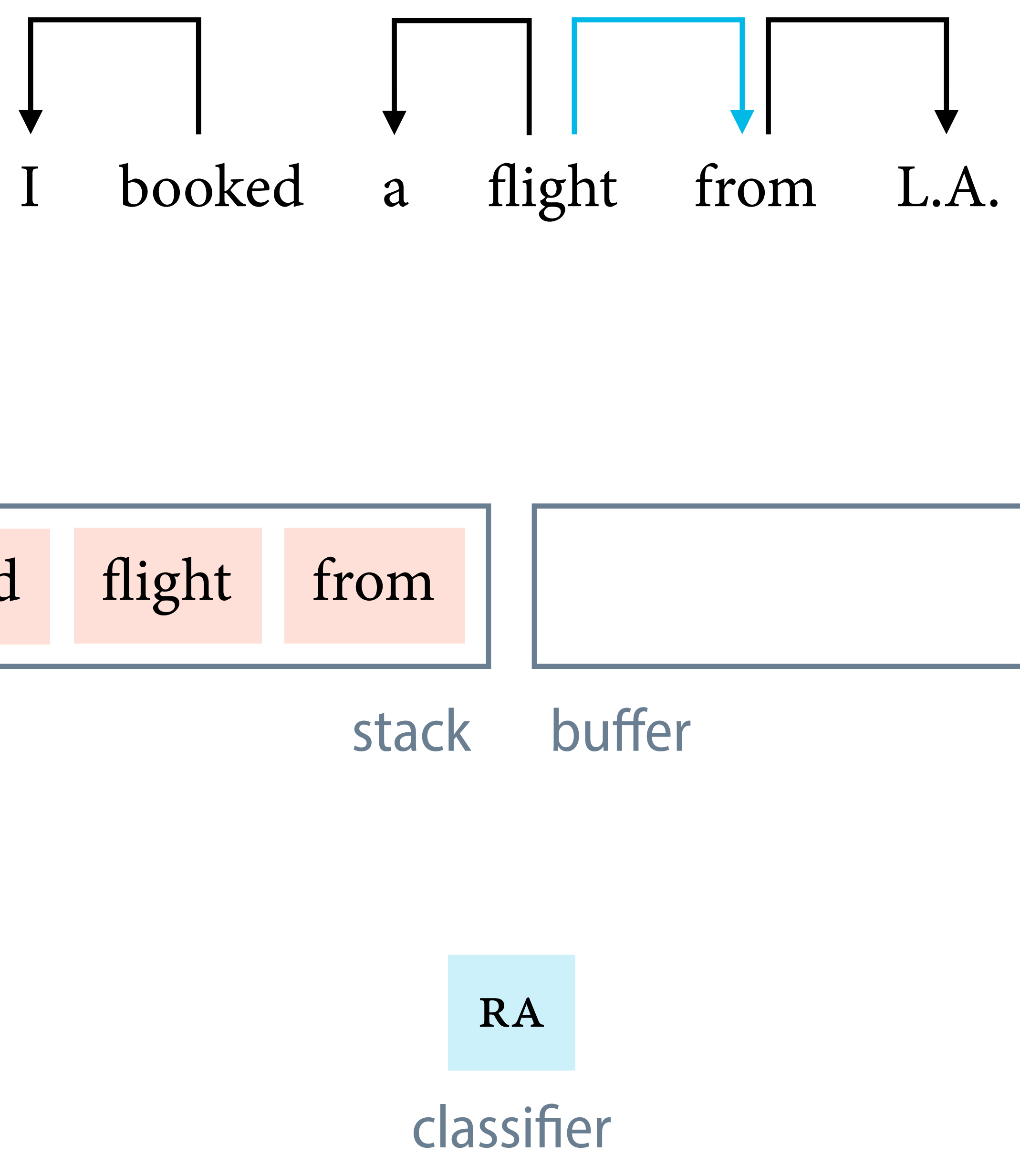


Transition-based dependency parsing, example

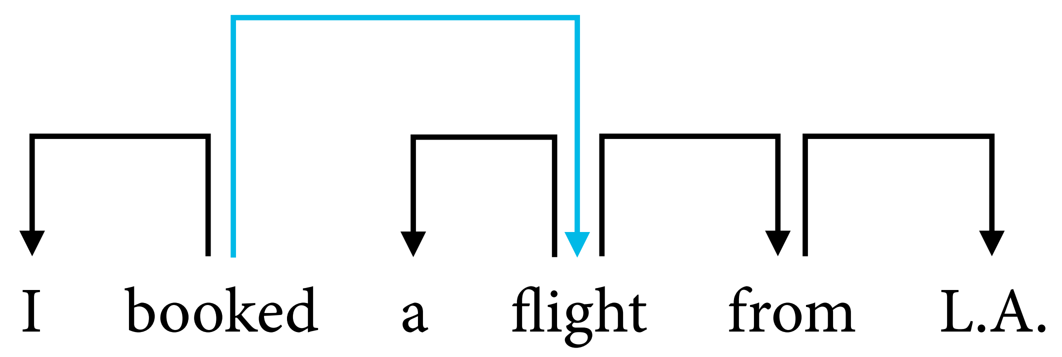


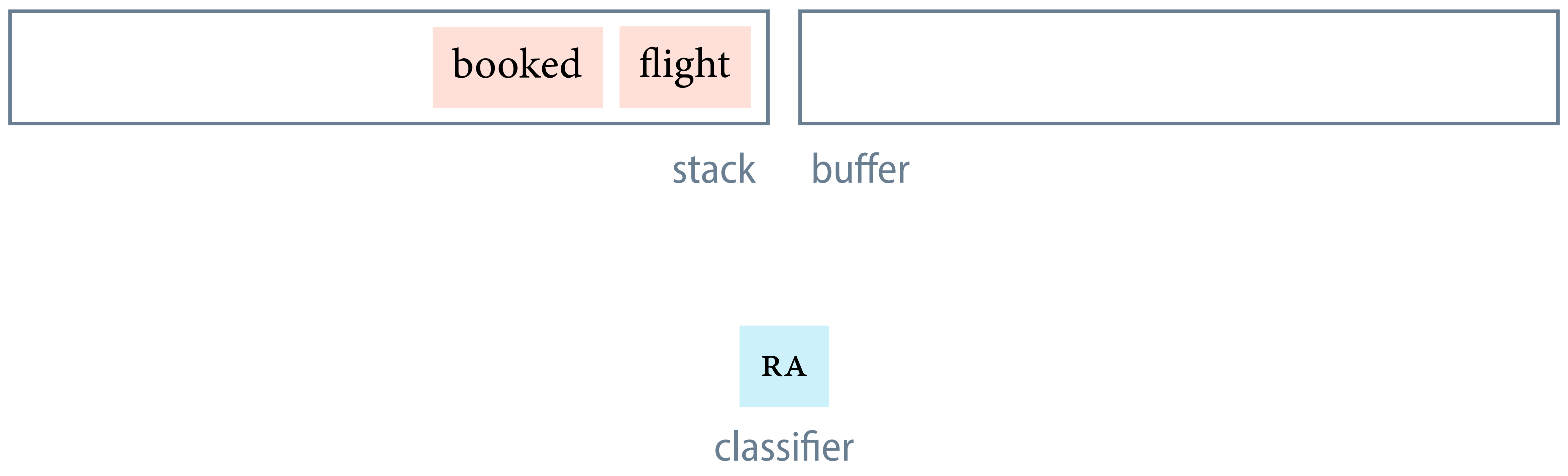


Transition-based dependency parsing, example

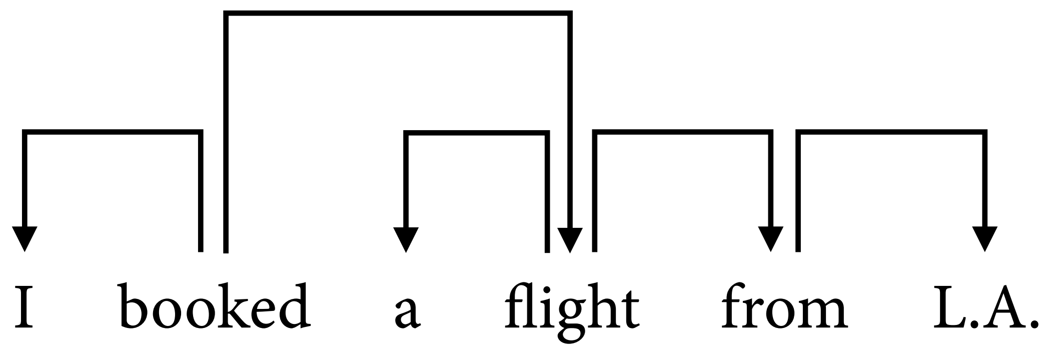


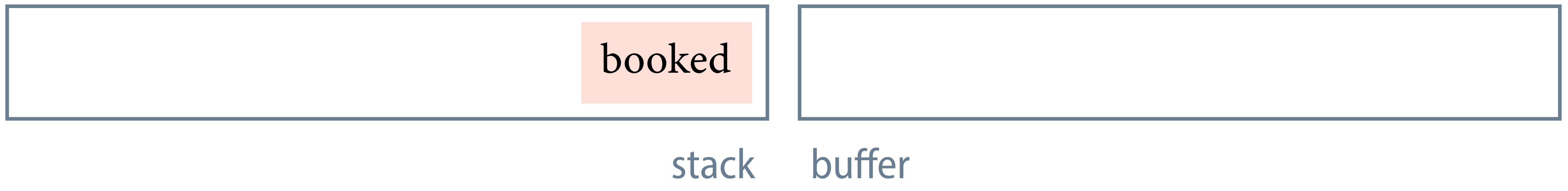
Transition-based dependency parsing, example





Transition-based dependency parsing, example





(terminal configuration)

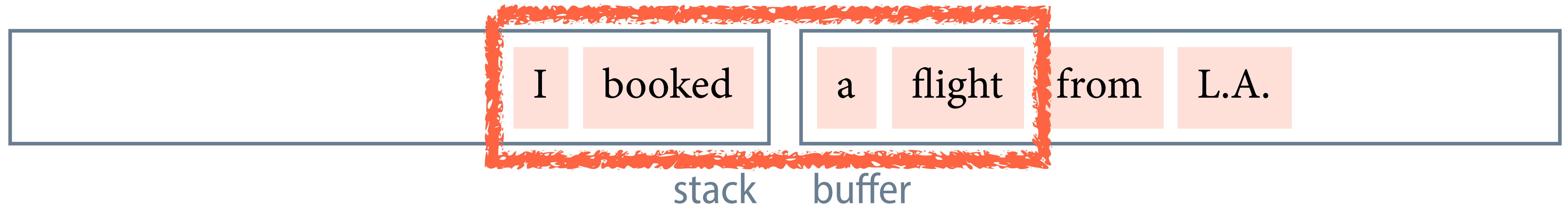
Features in transition-based dependency parsing

Features can be defined over

- the next words in the buffer
- the topmost words in the stack
- the partial dependency tree

Features in transition-based dependency parsing

I booked a flight from L.A.



Is 'booked' a verb? Can 'I' be a subject? Does 'booked' already have a subject?

Training transition-based dependency parsers

- To train a transition-based dependency parser, we need a treebank with dependency trees.
- In addition to that, we need an algorithm that tells us the gold-standard transition sequence for a tree in that treebank.

  oracle

Evaluating dependency parsers

- The **(unlabelled) attachment score** of a dependency parser is its accuracy at the task of predicting the correct head of each word.

  similar to the accuracy of a part-of-speech tagger

- When computing the **labelled attachment score**, we only give points if both the head *and* the dependency label are correct.

- The **exact match score** is the percentage of sentences in which *all* dependencies are correctly predicted.

  unlabelled = only heads, labelled = heads and labels

This lecture

- Introduction to syntactic analysis
- Parsing to phrase structure trees
  - Context-free grammars
  - Parsing with probabilistic context-free grammars
- Parsing to dependency trees
  - Transition-based dependency parsing