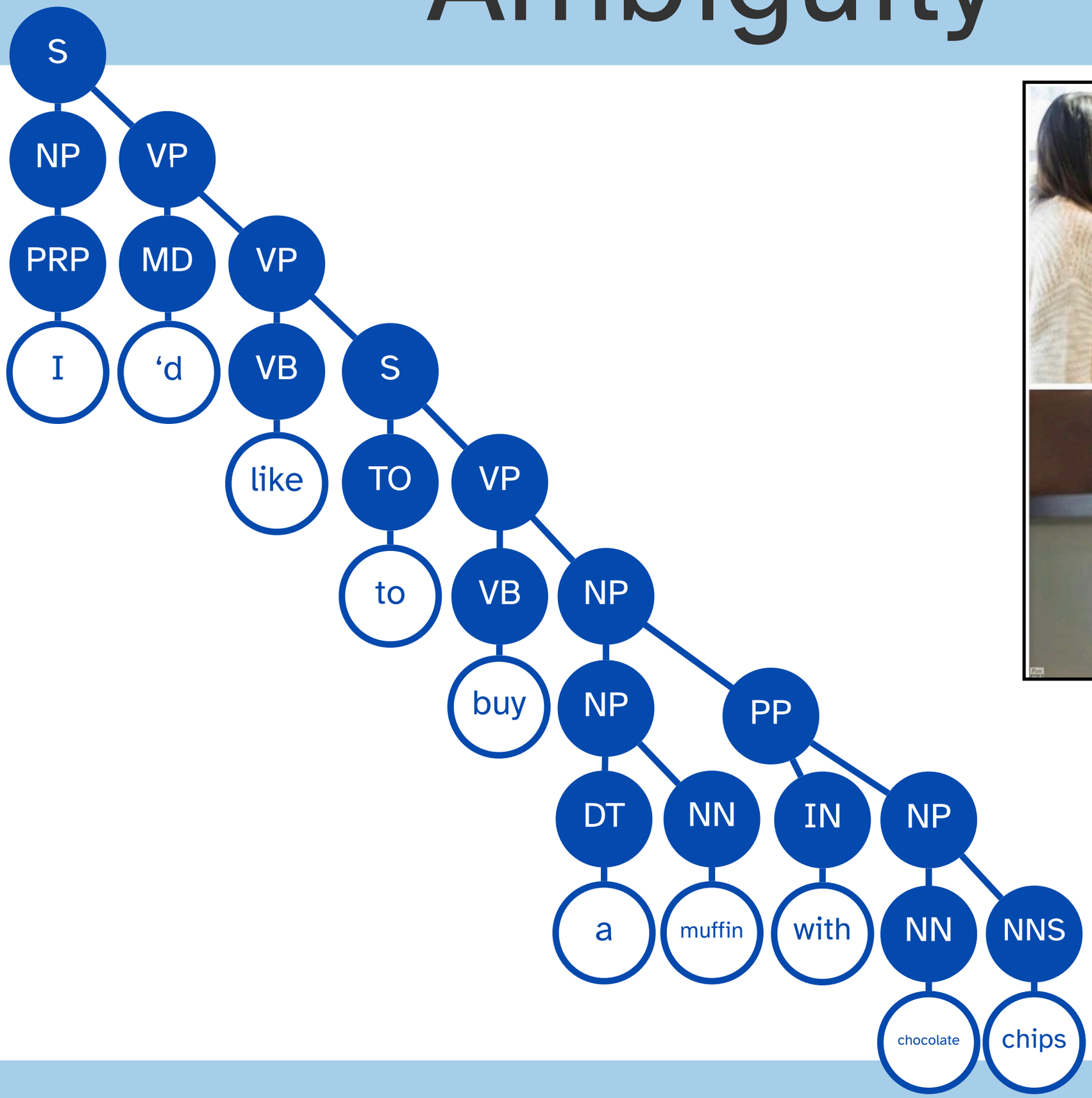
Linguistics: Syntax

EECS 183/283a: Natural Language Processing

Today

- Challenges for language modeling
  - Ambiguity
  - Grammaticality
- Formal languages
- Structures underlying natural language

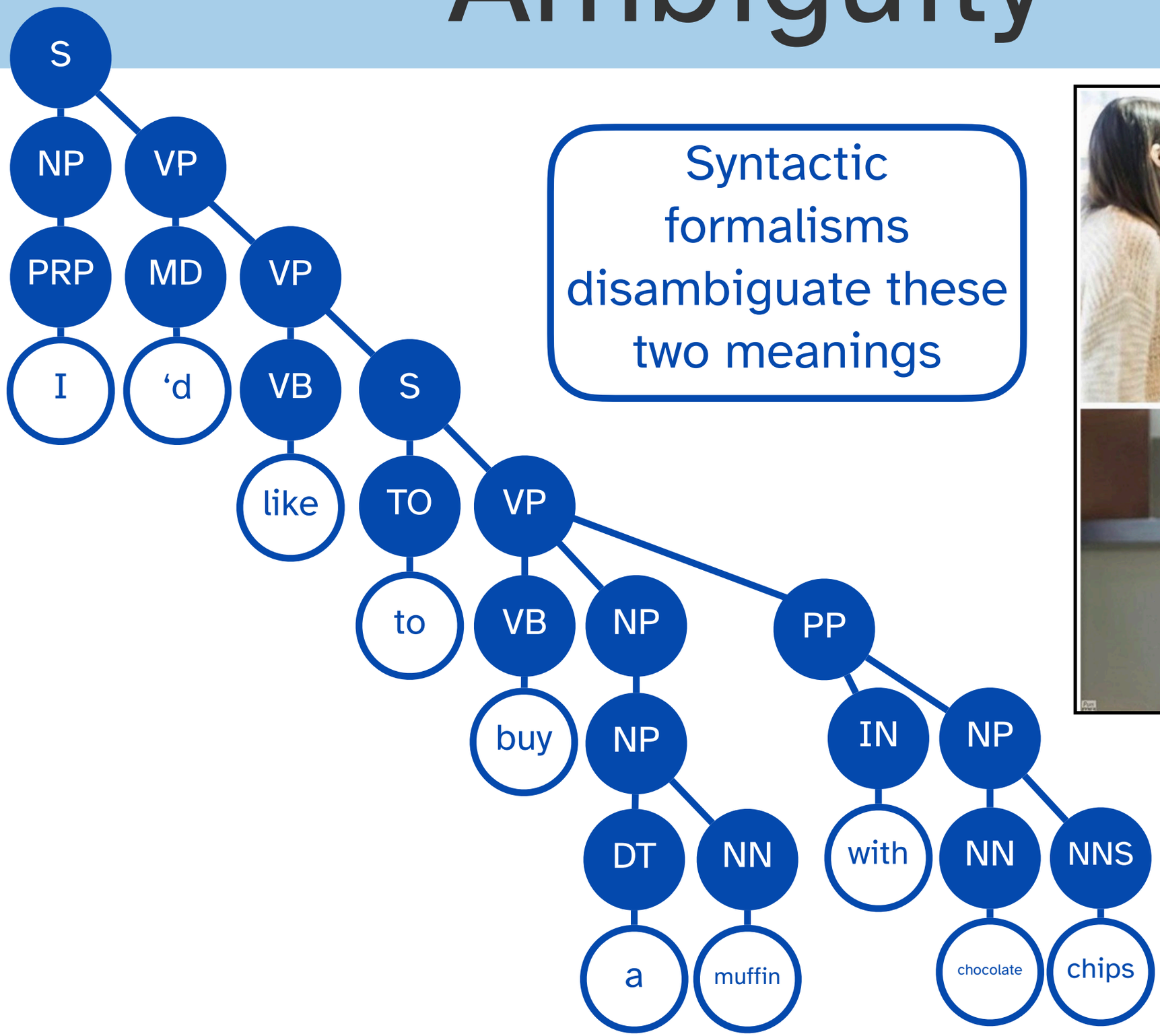
Ambiguity



example from Yoav Artzi

Preview: constituency parsing

Ambiguity



Syntactic formalisms disambiguate these two meanings

example from Yoav Artzi

Preview: constituency parsing

Grammaticality

- Does a sentence sound “right” according to a native speaker?

- Grammaticality judgments are not universal!
  - No such thing as “bad grammar” — disagreements over what feels grammatical or not are due to language variation
  - Grammatical sentences don’t need to be meaningful

Colorless green ideas sleep furiously

Chomsky 1957

Grammaticality

- Does a sentence sound “right” according to a native speaker?

- Grammaticality judgments are not universal!
  - No such thing as “bad grammar” — disagreements over what feels grammatical or not are due to language variation
  - Grammatical sentences don’t need to be meaningful

*Colorless green ideas sleep**s** furiously*

Chomsky 1957

Grammaticality

- Does a sentence sound “right” according to a native speaker?

- Grammaticality judgments are not universal!
  - No such thing as “bad grammar” — disagreements over what feels grammatical or not are due to language variation
  - Grammatical sentences don’t need to be meaningful

Green colorless *ideas sleep furiously*

Chomsky 1957

Grammaticality

- Why might something sound grammatical or not?
- Our main task: we have some language L, and we want to know whether a new sentence $x \in L$
- Should we represent L as a finite-sized set of possible sentences?
- What about a regular expression?

Regular Expressions

- Recall: (simplified) English syllable structure (C3)V(C4), with two types of sounds:
  - Consonants C: {p,b,t,d,k,g,m,n,ŋ,f,v,θ,ð,s,z,ʃ,ʒ,h,ɹ,l,j,w}
  - Vowels V: {i,u,ɛ,æ,ɪ,ɑ,ə,ʊ,ɔ,aɪ,oʊ,eɪ}
    - Allows us to generate “unattested” but still “accepted” syllables: θɹaɪm, skɹoʊp, fɹɪŋk
- Can syntax also be described with regular expressions?
  - How about {DET}{ADJ}*{NOUN}?

the agreement *an ill proposal* *a tiresome merciful teacher*

Regular Expressions

The cat that thinks the cow thinks
the rabbits hid are wrong. ✗

The rabbits hid.
DT N VB

Regex:

DT N VB

The cow thinks the rabbits hid.
DT N VB DT N VB (DT N VB)*

The cat that thinks the cow thinks the rabbits hid is wrong.
DT N IN VB DT N VB DT N VB VB JJ

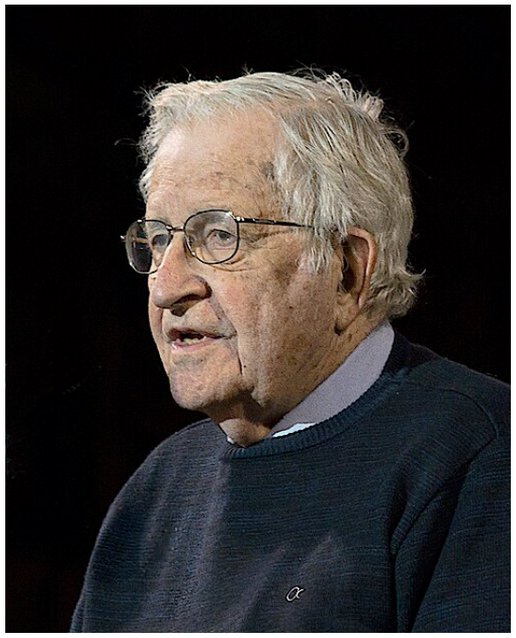
DT N IN (DT N VB)* VB JJ?

Is there another formalism that tells us whether
a string is “accepted” in a language or not?

Context-Free Grammars

- CFGs offer arbitrary expressivity through recursive structure

- You might have seen these before in programming languages classes — basically, most programming languages are CFGs



Σ, Wikpiedia

- CFGs were actually first described formally by Noam Chomsky (b. 1928), but the recursive structure of natural language was described by many linguists, including Pāṇini (~1-5th century BCE)

Context-Free Grammars

- Set of nonterminal symbols
- Set of terminal symbols (wordtypes)
- Set of production rules defining how nonterminal symbols could be expressed via the composition of other nonterminal and terminal symbols

Nonterminal symbols

DT N VB JJ IN

S NP VP SBAR

Terminal symbols (vocabulary)

are, cat, cow, hid, is, rabbits, that, the, thinks, wrong, ...

Production rules

DT → {*the, a, an, ...*}

N → {*cat, cow, rabbits, dogs, ...*}

VB → {*hid, is, thinks, was, are, ...*}

JJ → {*wrong, right, blue, red, ...*}

IN → {*that, in, of, because, ...*}

NP → {DT N, NP IN VP}

VP → {VB, VB JJ, VB S}

S → {NP VP, VP}

Context-Free Grammars

The cat that thinks the cow thinks the rabbits hid is wrong.
DT DT DT

Nonterminal symbols

DT N VB JJ IN

S NP VP SBAR

Terminal symbols (vocabulary)

are, cat, cow, hid, is, rabbits, that, the, thinks, wrong, ...

Production rules

DT → {*the, a, an, ...*}

N → {*cat, cow, rabbits, dogs, ...*}

VB → {*hid, is, thinks, was, are, ...*}

JJ → {*wrong, right, blue, red, ...*}

IN → {*that, in, of, because, ...*}

NP → {DT N, NP IN VP}

VP → {VB, VB JJ, VB S}

S → {NP VP, VP}

Context-Free Grammars

The cat that thinks the cow thinks the rabbits hid is wrong.
DT N DT N DT N

Nonterminal symbols

DT N VB JJ IN

S NP VP SBAR

Terminal symbols (vocabulary)
are, cat, cow, hid, is, rabbits, that, the, thinks, wrong, …

Production rules

DT → {*the, a, an, …*}
N → {*cat, cow, rabbits, dogs, …*}
VB → {*hid, is, thinks, was, are, …*}
JJ → {*wrong, right, blue, red, …*}
IN → {*that, in, of, because, …*}
NP → {DT N, NP IN VP}
VP → {VB, VB JJ, VB S}
S → {NP VP, VP}

Context-Free Grammars

The cat that thinks the cow thinks the rabbits hid is wrong.
DT N VB DT N VB DT N VB VB

Nonterminal symbols

DT N VB JJ IN

S NP VP SBAR

Terminal symbols (vocabulary)
are, cat, cow, hid, is, rabbits, that, the, thinks, wrong, ...

Production rules

DT → {*the, a, an, ...*}
N → {*cat, cow, rabbits, dogs, ...*}
VB → {*hid, is, thinks, was, are, ...*}
JJ → {*wrong, right, blue, red, ...*}
IN → {*that, in, of, because, ...*}
NP → {DT N, NP IN VP}
VP → {VB, VB JJ, VB S}
S → {NP VP, VP}

Context-Free Grammars

The cat that thinks the cow thinks the rabbits hid is wrong.

DT N VB DT N VB DT N VB VB JJ

Nonterminal symbols

DT N VB JJ IN

S NP VP SBAR

Terminal symbols (vocabulary)

are, cat, cow, hid, is, rabbits, that, the, thinks, wrong, ...

Production rules

DT → {*the, a, an, ...*}

N → {*cat, cow, rabbits, dogs, ...*}

VB → {*hid, is, thinks, was, are, ...*}

JJ → {*wrong, right, blue, red, ...*}

IN → {*that, in, of, because, ...*}

NP → {DT N, NP IN VP}

VP → {VB, VB JJ, VB S}

S → {NP VP, VP}

Context-Free Grammars

The cat that thinks the cow thinks the rabbits hid is wrong.

DT N IN VB DT N VB DT N VB VB JJ

Nonterminal symbols

DT N VB JJ IN

S NP VP SBAR

Terminal symbols (vocabulary)

are, cat, cow, hid, is, rabbits, that, the, thinks, wrong, ...

Production rules

DT → {*the, a, an, ...*}

N → {*cat, cow, rabbits, dogs, ...*}

VB → {*hid, is, thinks, was, are, ...*}

JJ → {*wrong, right, blue, red, ...*}

IN → {*that, in, of, because, ...*}

NP → {DT N, NP IN VP}

VP → {VB, VB JJ, VB S}

S → {NP VP, VP}

Context-Free Grammars

The cat	that	thinks	the cow	thinks	the rabbits	hid	is	wrong.
DT N	IN	VB	DT N	VB	DT N	VB	VB	JJ
NP			**NP**		**NP**			

Nonterminal symbols

DT N VB JJ IN

S NP VP

Production rules

NP → {DT N, NP IN VP}
VP → {VB, VB JJ, VB S}
S → {NP VP, VP}

Context-Free Grammars

The cat	that	thinks	the cow	thinks	the rabbits	hid	is	wrong.
DT N	IN	VB	DT N	VB	DT N	VB	VB	JJ
NP			**NP**		**NP**	**VP**		

Nonterminal symbols

DT N VB JJ IN

S NP VP

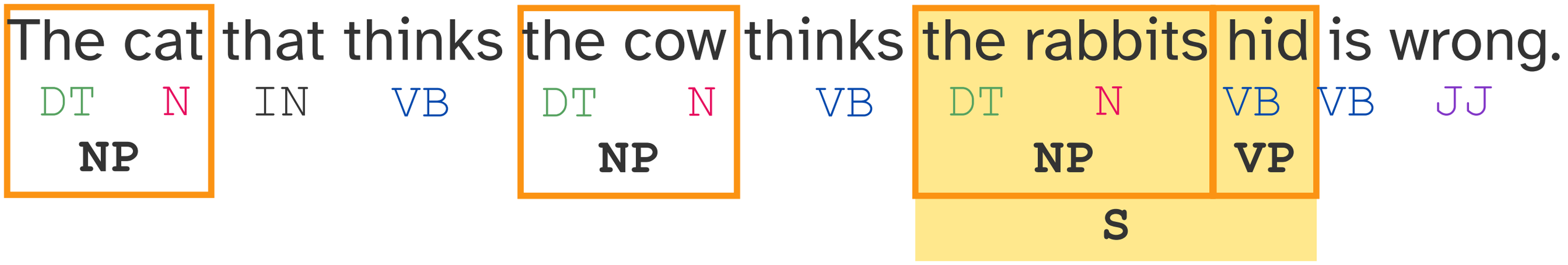
Production rules

NP → {DT N, NP IN VP}

VP → {VB, VB JJ, VB S}

S → {NP VP, VP}

Context-Free Grammars



Nonterminal symbols

DT N VB JJ IN

S NP VP

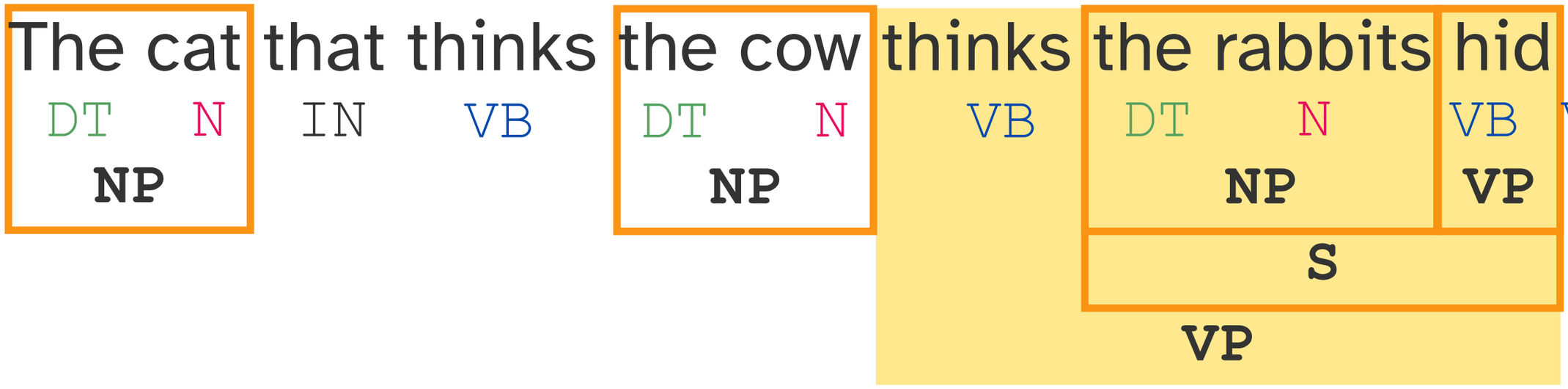
Production rules

NP → {DT N, NP IN VP}

VP → {VB, VB JJ, VB S}

S → {NP VP, VP}

Context-Free Grammars



Nonterminal symbols

DT N VB JJ IN

S NP VP

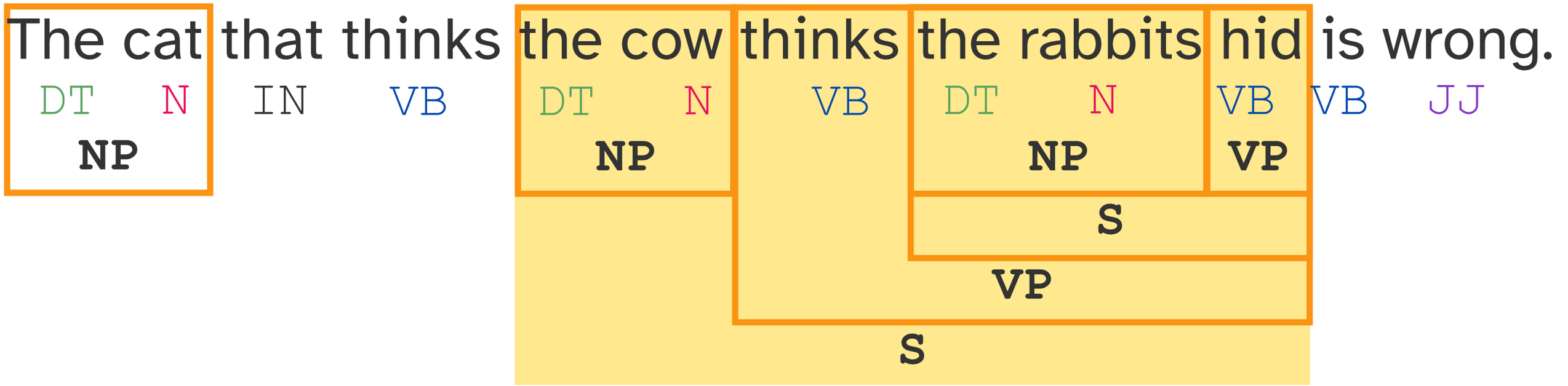
Production rules

NP → {DT N, NP IN VP}

VP → {VB, VB JJ, VB S}

S → {NP VP, VP}

Context-Free Grammars



Nonterminal symbols

DT N VB JJ IN

S NP VP

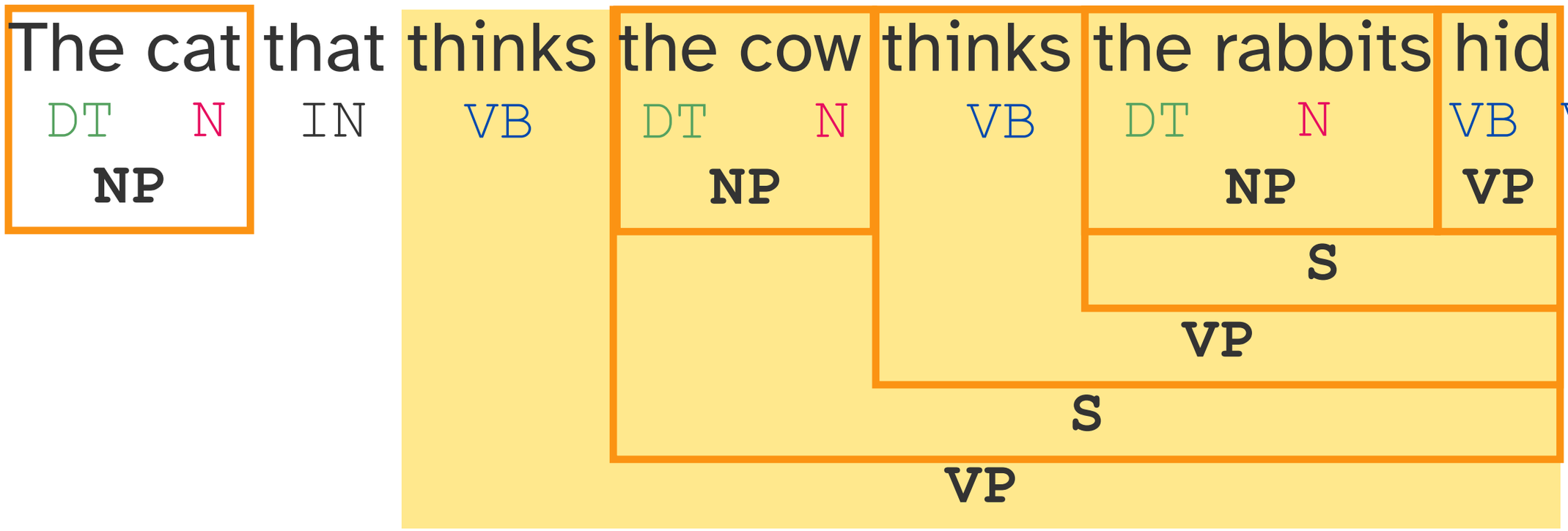
Production rules

NP → {DT N, NP IN VP}

VP → {VB, VB JJ, VB S}

S → {NP VP, VP}

Context-Free Grammars



Nonterminal symbols

DT N VB JJ IN

S NP VP

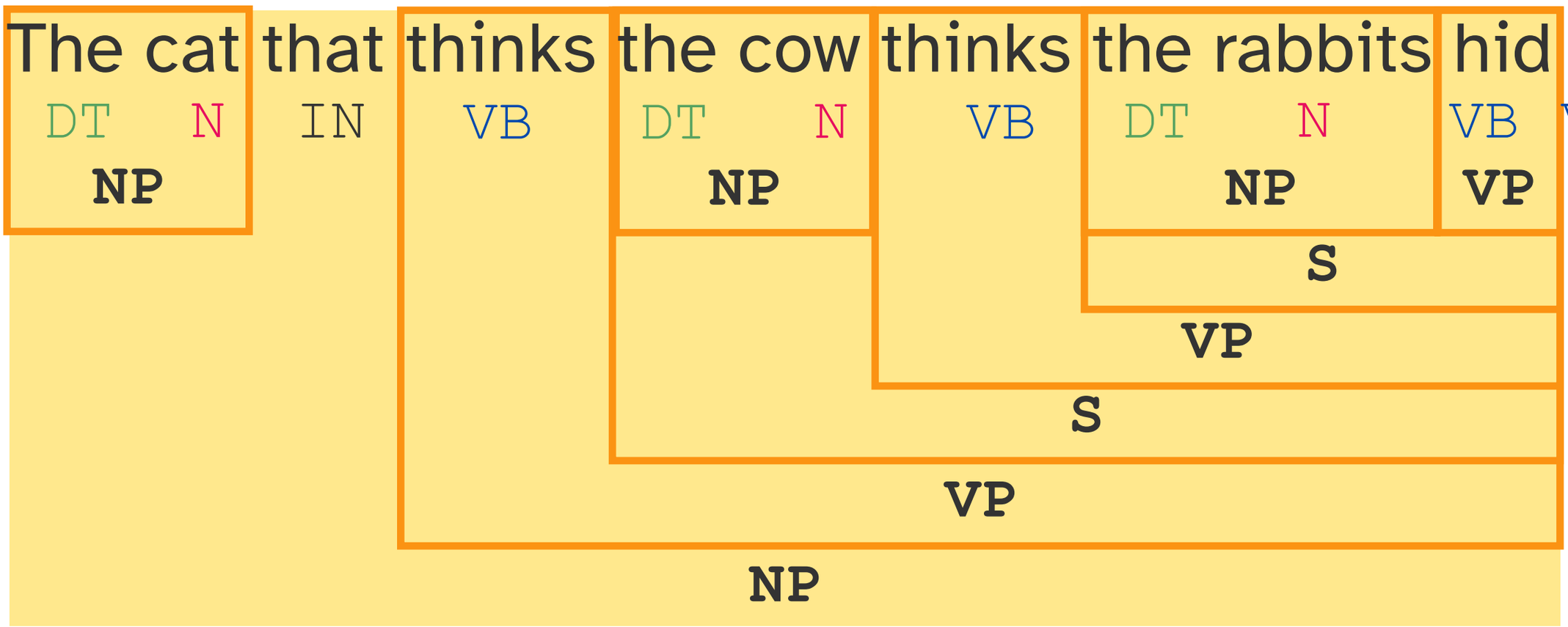
Production rules

NP → {DT N, NP IN VP}

VP → {VB, VB JJ, VB S}

S → {NP VP, VP}

Context-Free Grammars



Nonterminal symbols

DT N VB JJ IN

S NP VP

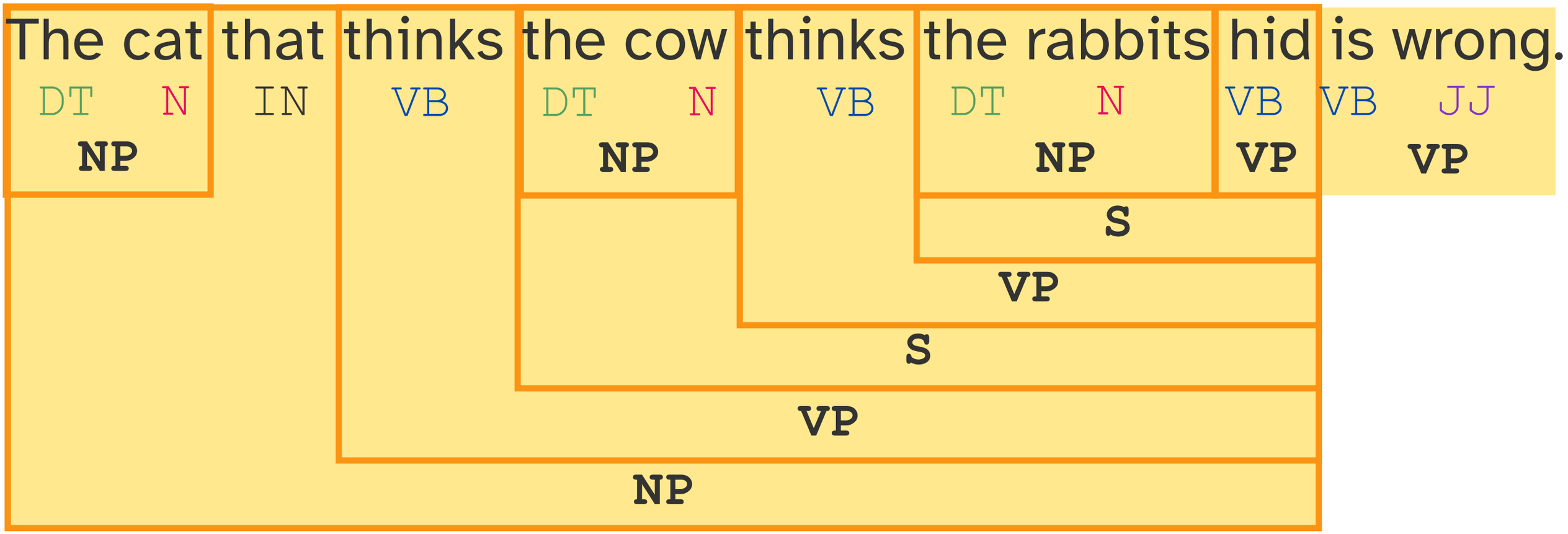
Production rules

NP → {DT N, NP IN VP}

VP → {VB, VB JJ, VB S}

S → {NP VP, VP}

Context-Free Grammars



Nonterminal symbols

DT N VB JJ IN

S NP VP

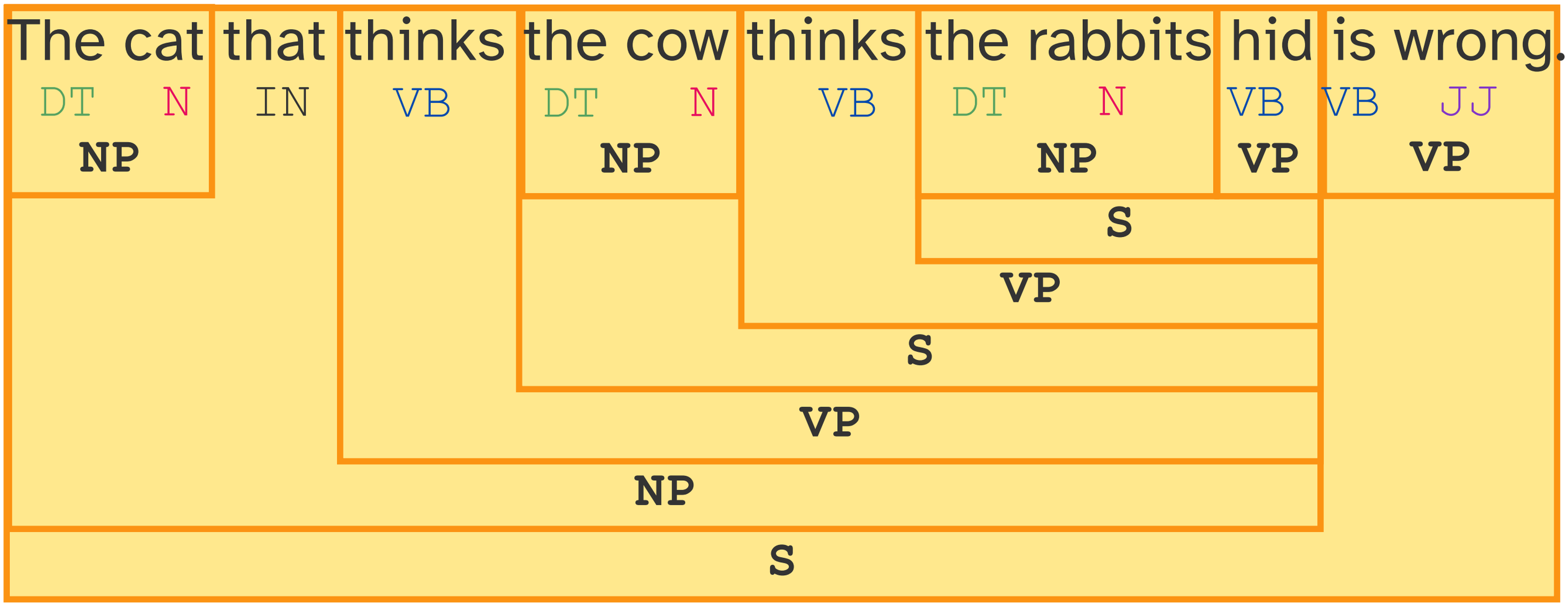
Production rules

NP → {DT N, NP IN VP}

VP → {VB, VB JJ, VB S}

S → {NP VP, VP}

Context-Free Grammars



NP is singular, so its corresponding VP should be too

Nonterminal symbols

DT N VB JJ IN

S NP VP

Production rules

NP → {DT N, NP IN VP}

VP → {VB, VB JJ, VB S}

S → {NP VP, VP}

Generating from a Grammar

Nonterminal symbols

DT N VB JJ IN

S NP VP SBAR

Terminal symbols (vocabulary)

are, cat, cow, hid, is, rabbits, that, the, thinks, wrong, …

Production rules

DT → {*the, a, an, …*}

N → {*cat, cow, rabbits, dogs, …*}

VB → {*hid, is, thinks, was, are, …*}

JJ → {*wrong, right, blue, red, …*}

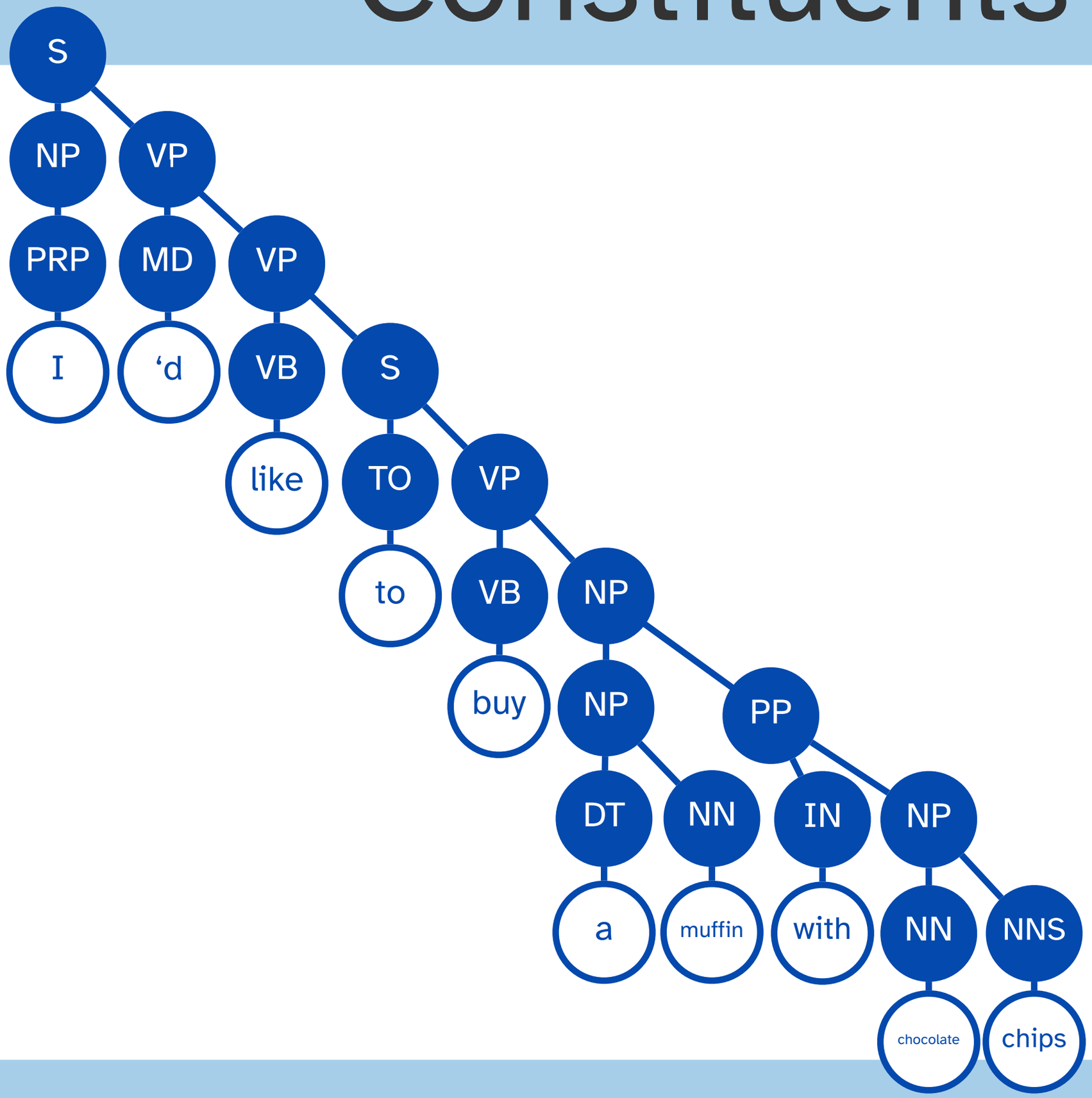
IN → {*that, in, of, because, …*}

NP → {DT N, NP IN VP}

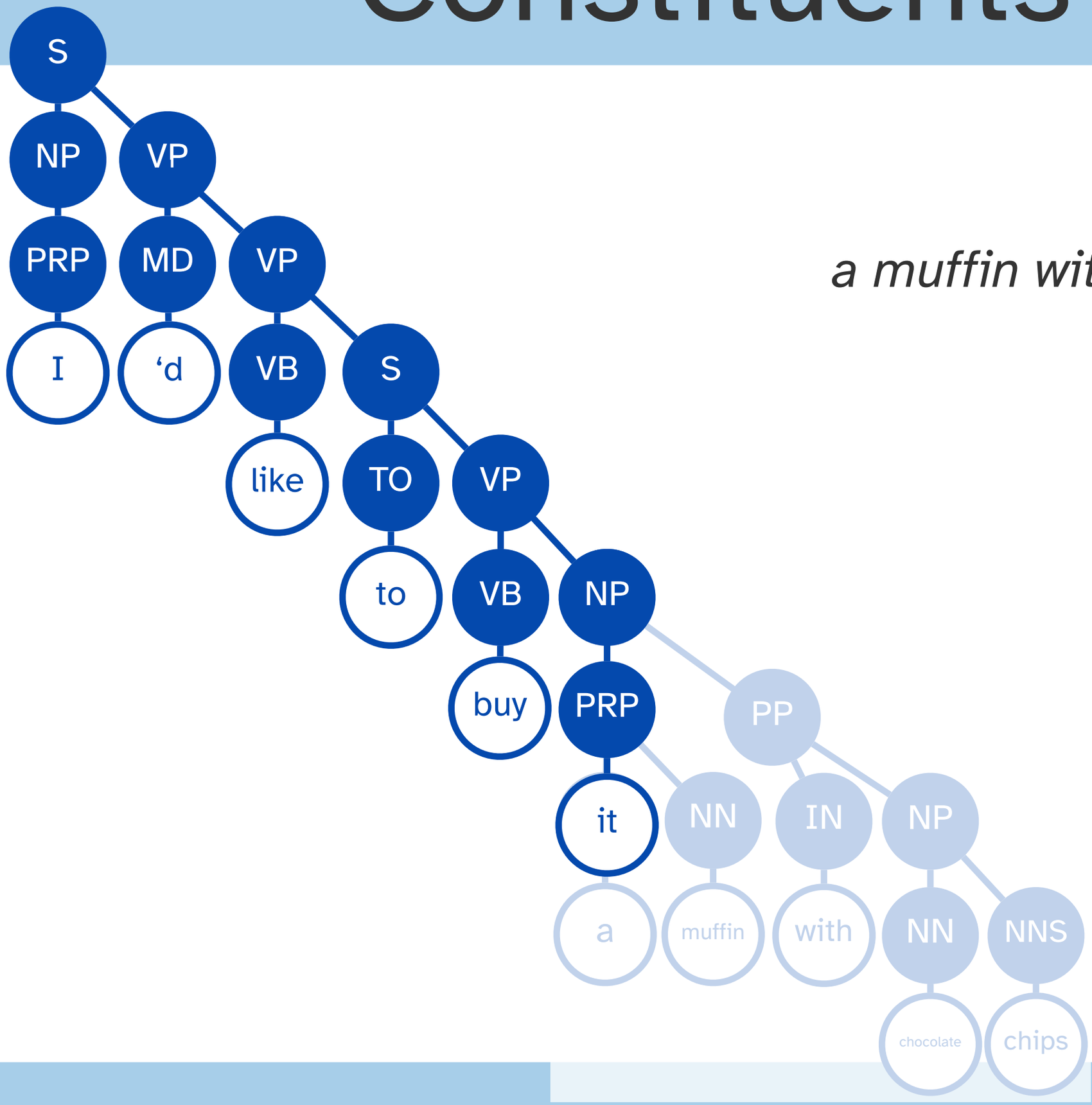
VP → {VB, VB JJ, VB S}

S → {NP VP, VP}

Constituents



Constituents

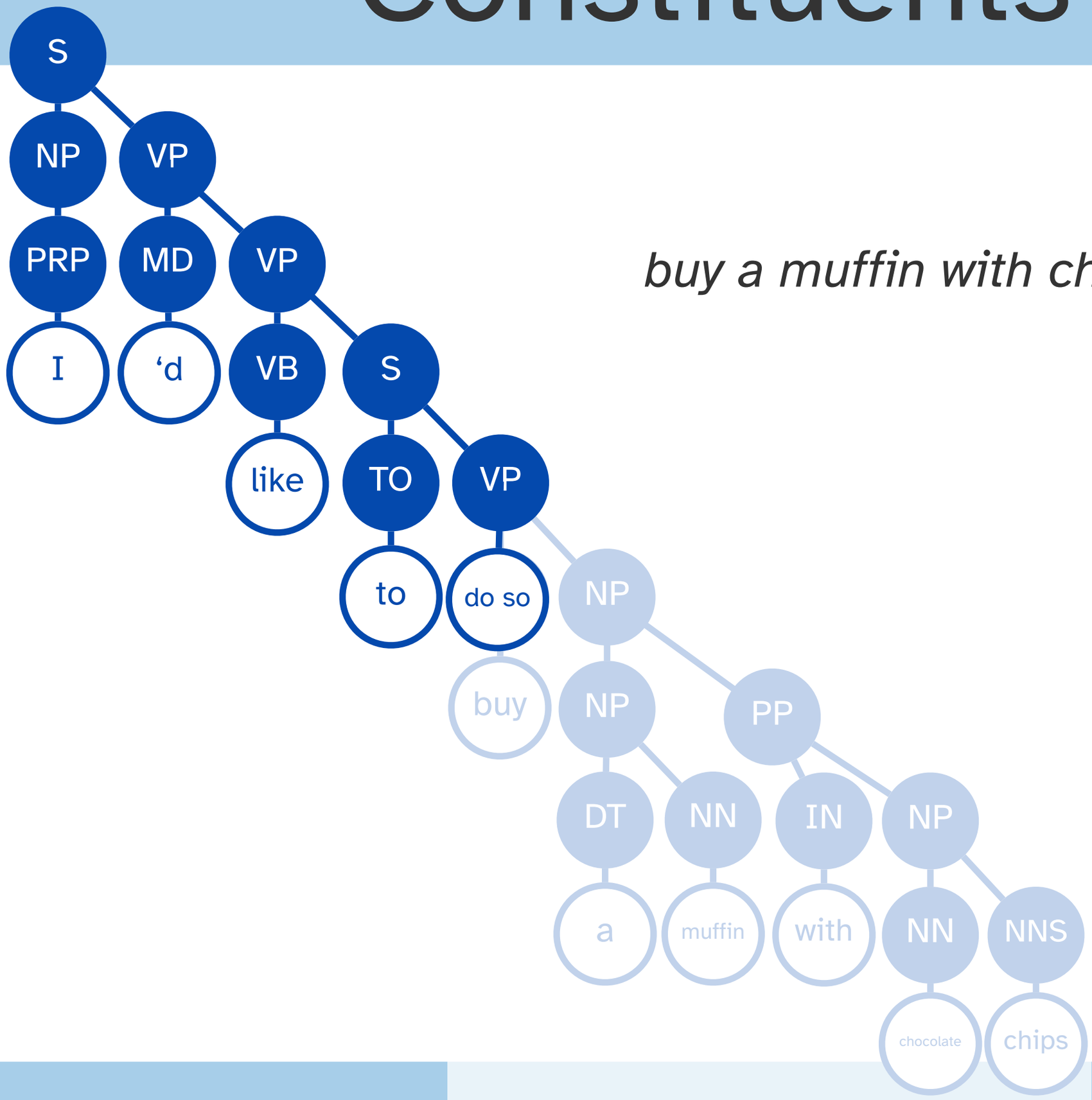


Substitution:

NP → NP PP **or** PRP

a muffin with chocolate chips → *it*

Constituents

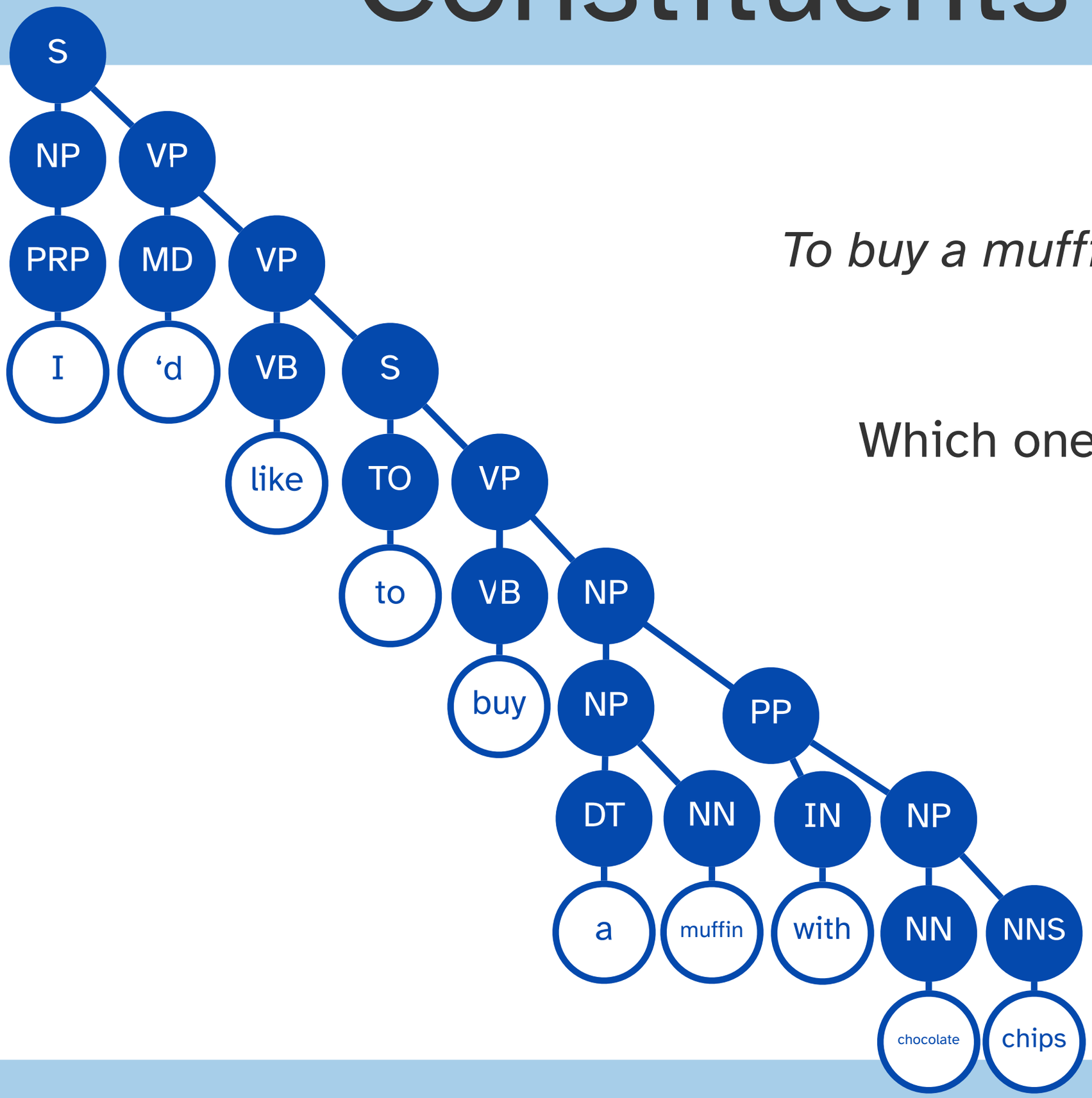


Substitution:

VP → *do so*

buy a muffin with chocolate chips → *do so*

Constituents

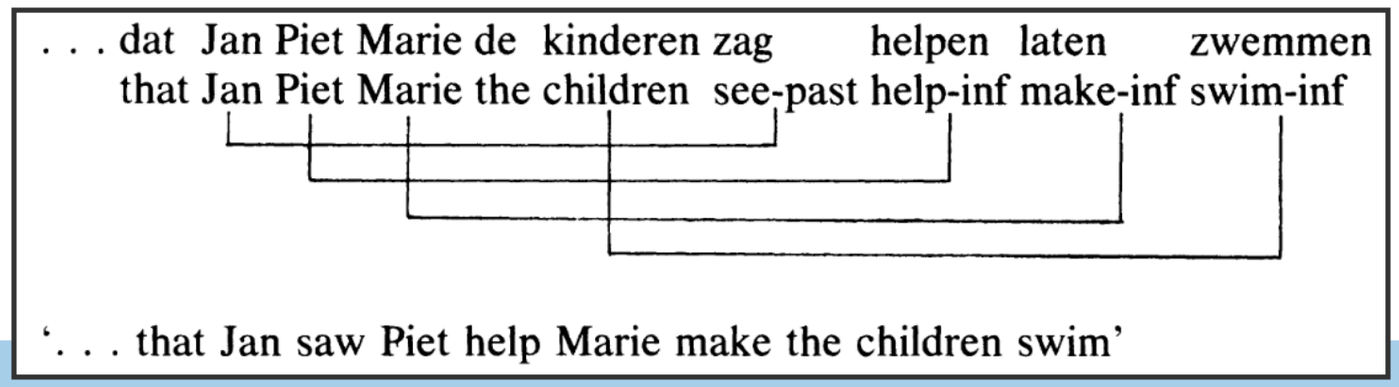


Answer ellipsis:
What would I like?
To buy a muffin with chocolate chips

Which one would you like to buy?
With chocolate chips

Constituents

- CFG is often called phrase structure or constituency grammar
- Each production rule describes a **constituent**
- Constituent constructions are independent of one another (this is why the grammar is context-free)
- Augmenting a CFG with agreement (e.g., distinguishing plural vs. singular NPs and plural vs. singular VPs plural) means it is no longer context-free
- Also, some languages aren't even context-free, not even considering agreement:



Kaplan et al. 1982

Probabilistic CFG

Production rules

augmented with probabilities

DT → {*the*, *a*, *an*, …} 0.5 0.2 0.2

N → {*cat*, *cow*, *rabbits*, *dogs*, …} 0.02 0.01 0.01 0.02

VB → {*hid*, *is*, *thinks*, *was*, *are*, …} 0.01 0.5 0.05 0.3 0.2

JJ → {*wrong*, *right*, *blue*, *red*, …} 0.05 0.1 0.05 0.05

IN → {*that*, *in*, *of*, *because*, …} 0.4 0.3 0.2 0.05

NP → {DT N, NP IN VP, …} 0.8 0.1

VP → {VB, VB JJ, VB S, …} 0.3 0.2 0.1

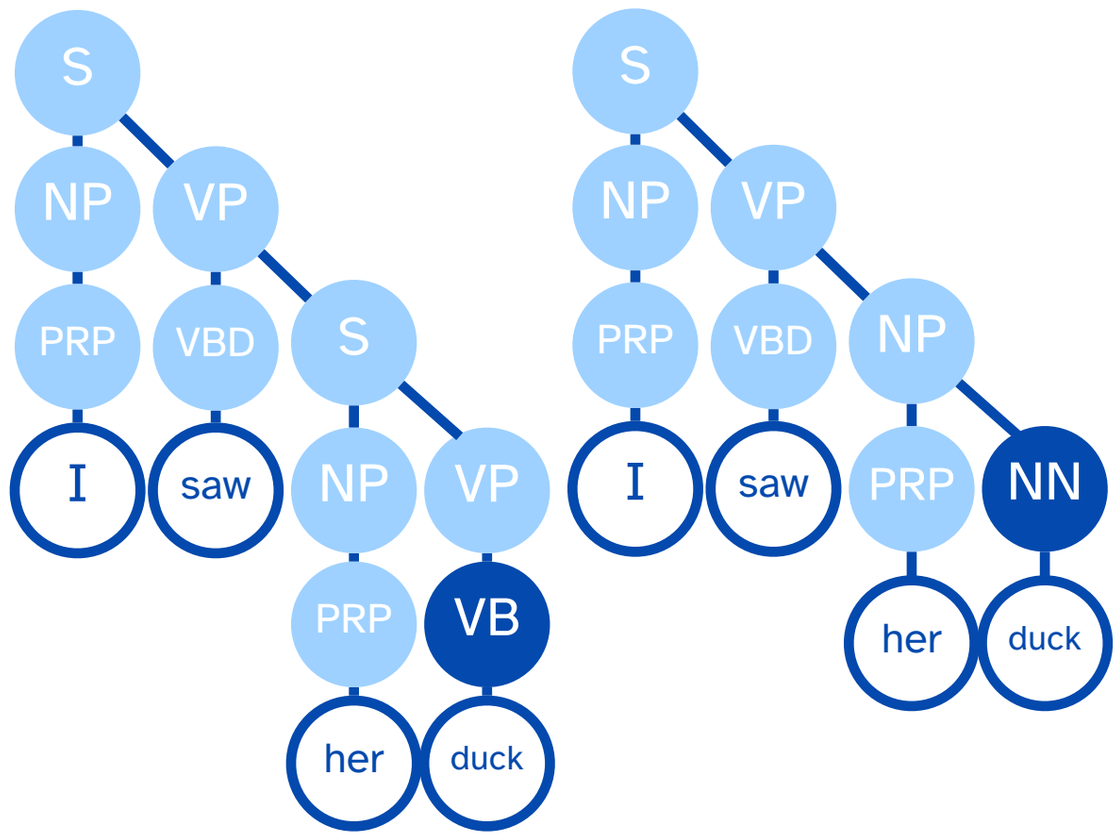
S → {NP VP, VP, …} 0.7 0.05

$$p(\text{rule}|\text{nonterminal})$$

learn from data

syntactic parsing: given a PCFG, which rule applications generated our sentence?

Why is this hard?



Combinatory Categorial Grammar (CCG)

- Another way of representing a constituency grammar: bottom-up

- Elements of a CCG:
  - Lexical items (wordtypes)
  - Each paired with a syntactic type (≈ nonterminal or composition thereof)

$\text{the} : NP/N$	$\text{dog} : N$	$\text{John} : NP$	$\text{bit} : (S\backslash NP)/NP$
If a Noun appears to the right, then it creates a Noun Phrase	Noun	Noun Phrase	If a Noun Phrase appears to the right, then it creates an element with the type S \ NP
NP → DT N DT → *the*	N → *dog*	NP → PRO PRO → *John*	If an NP appears to the left of that element, it creates a Sentence

Example from Wikipedia

Combinatory Categorial Grammar (CCG)

- Another way of representing a constituency grammar: bottom-up

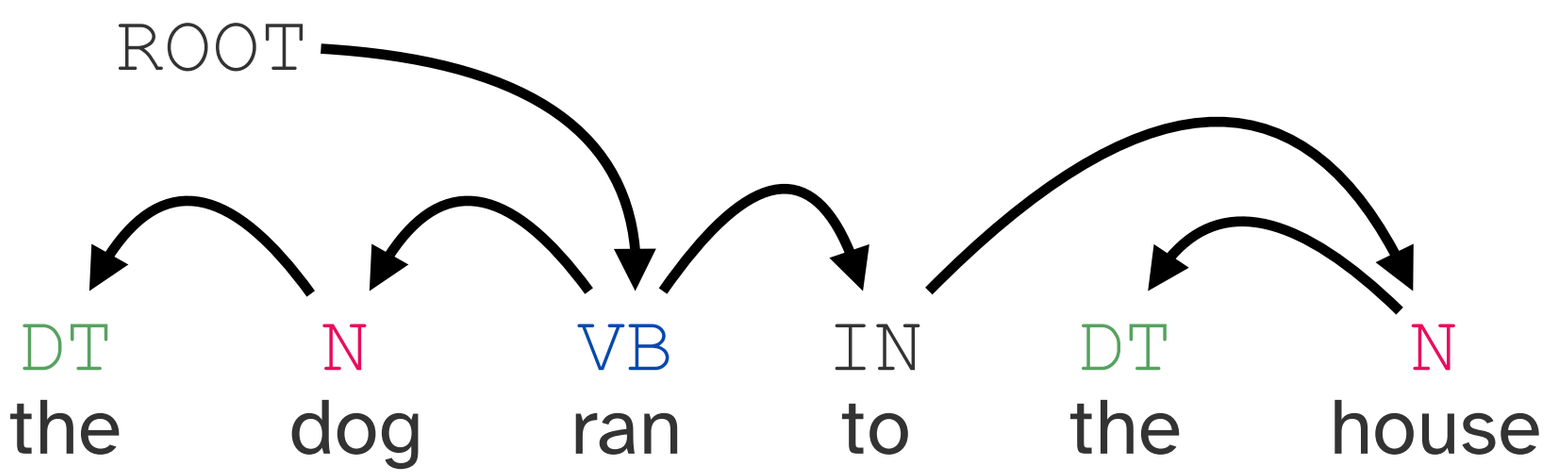
- Elements of a CCG:
  - Lexical items (wordtypes)
  - Each paired with a syntactic type (≈ nonterminal or composition thereof)

the : NP/N	dog : N	John : NP	bit : $(S\backslash NP)/NP$
the	dog	bit	John

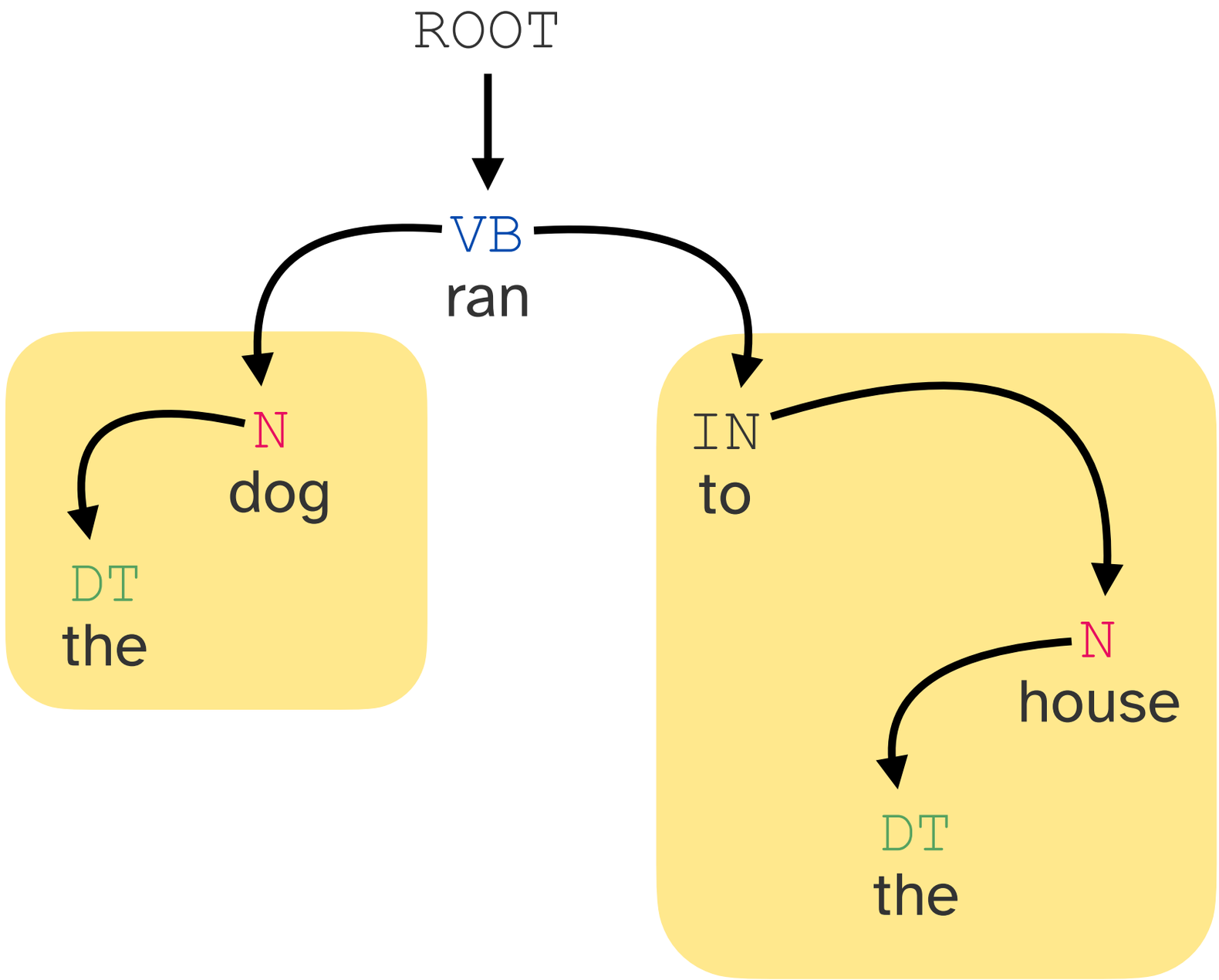
Example from Wikipedia

Dependency Grammar

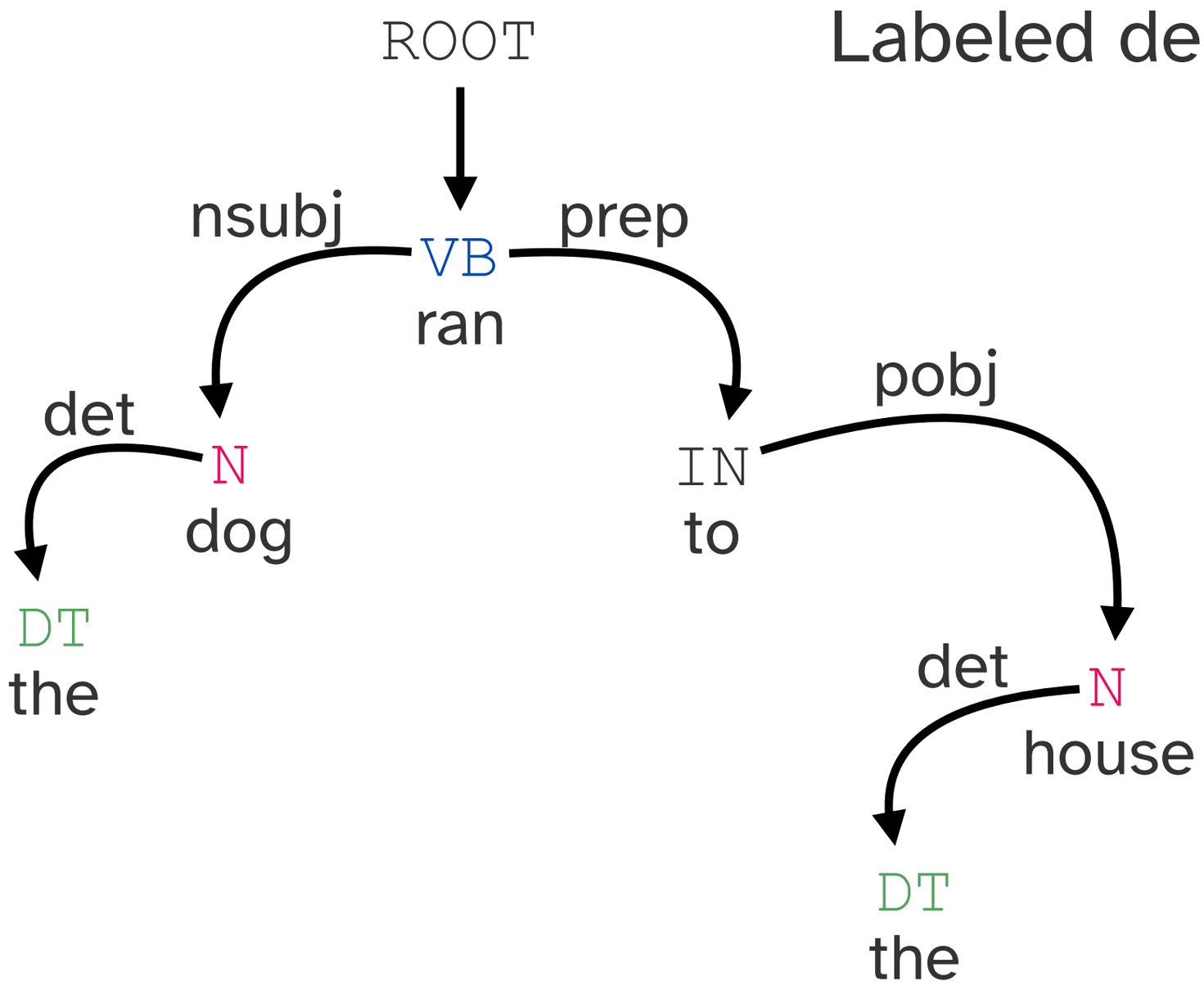
- A dependency grammar includes:
  - Terminal symbols (wordtypes)
  - Parts of speech
  - Some constraints on which parts of speech can be attached to other parts of speech



Dependency Grammar

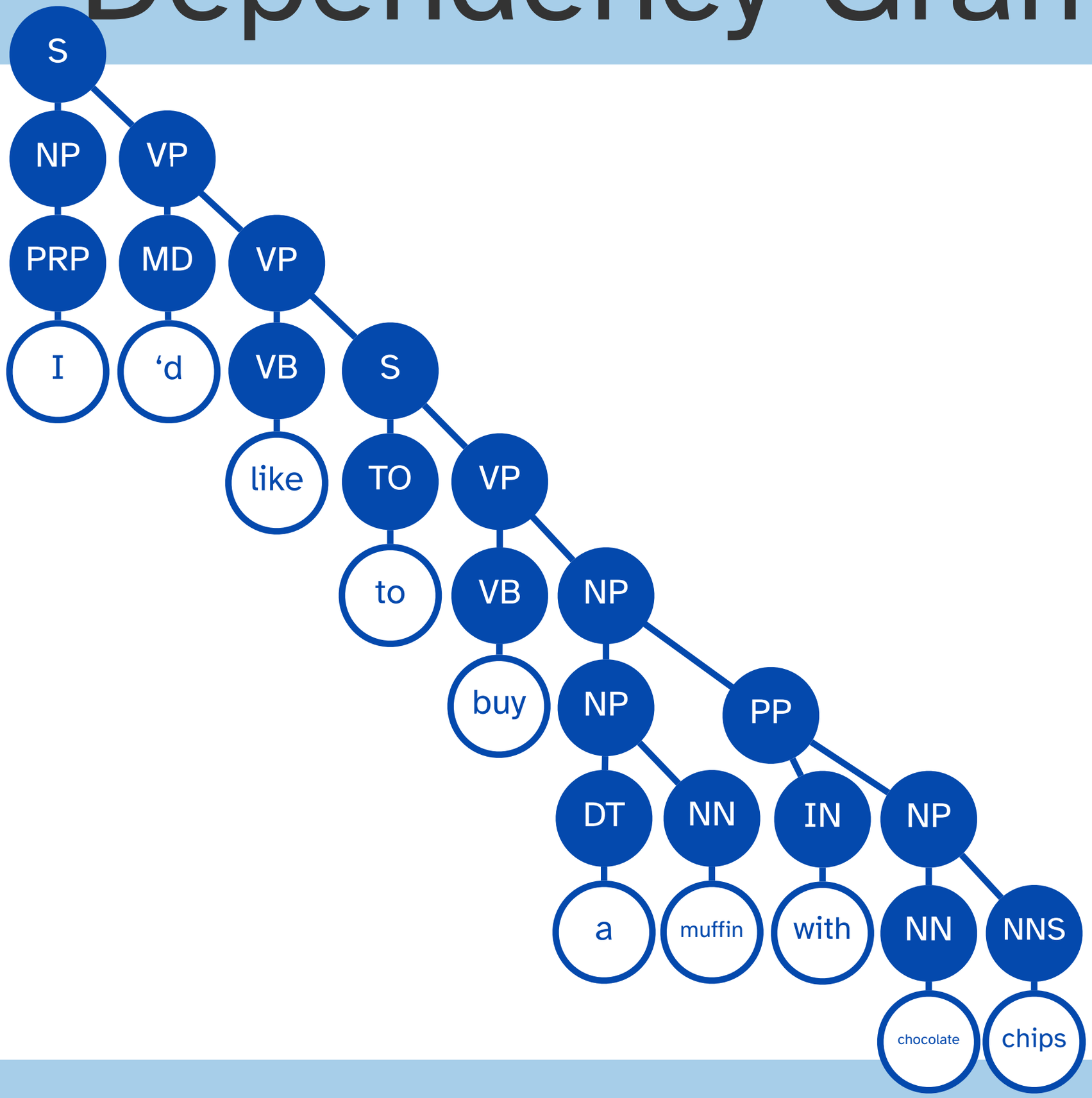


Dependency Grammar

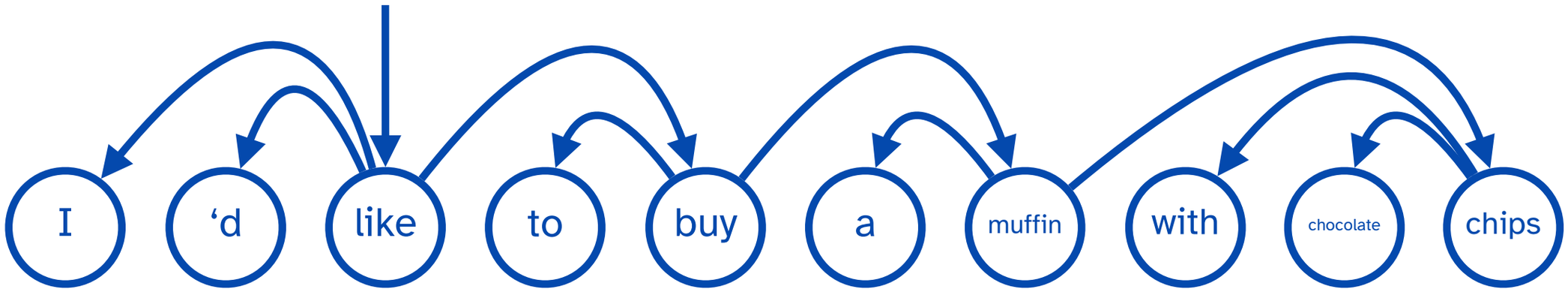


Labeled dependencies

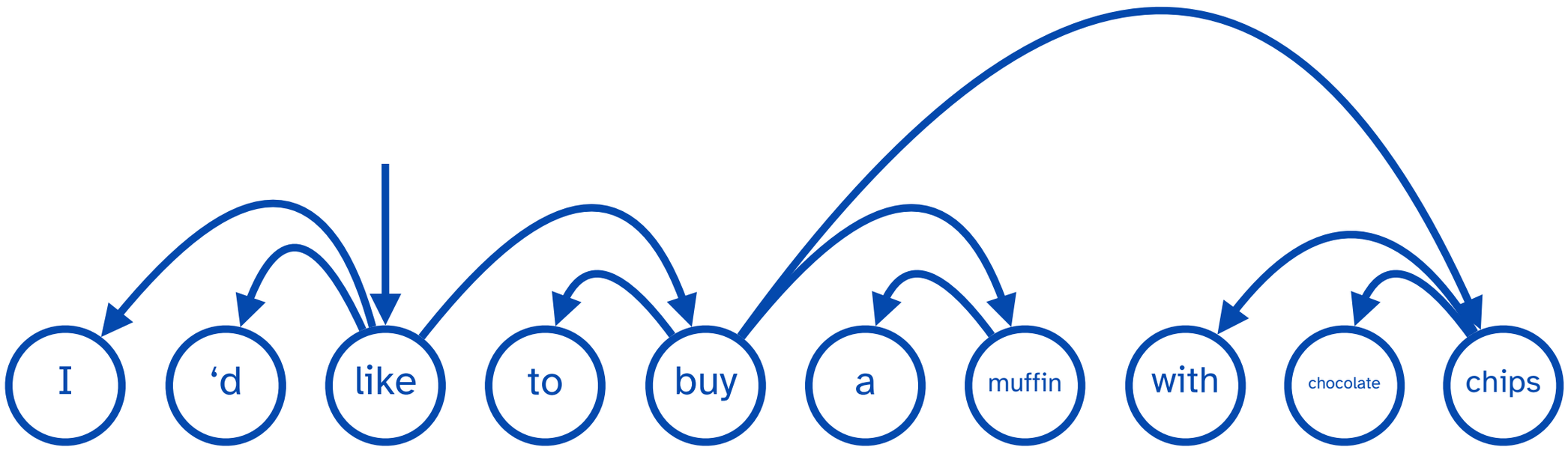
Dependency Grammar



Dependency Grammar



Dependency Grammar



Universal Dependencies

Huge resource of grammar annotations across ~150 languages

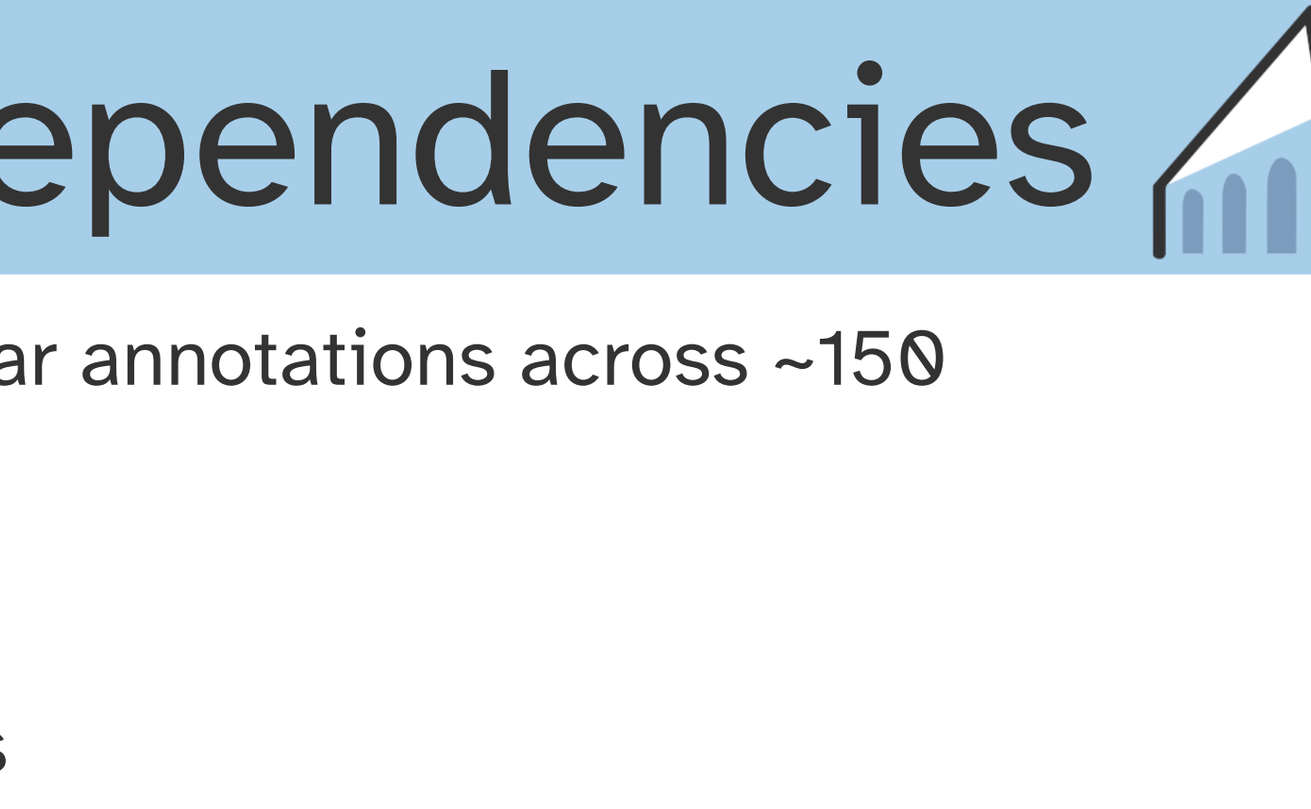
- Parts of speech
- Morphological features
- Syntactic dependency parses

English

Bulgarian

Czech

Swedish



https://universaldependencies.org/