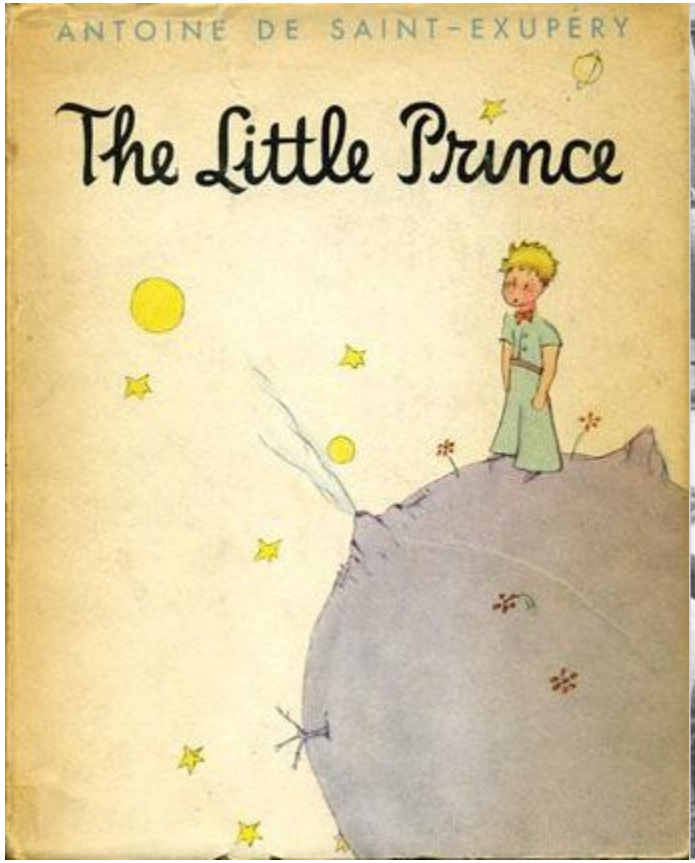
CS440/ECE448 Lecture 17: Natural Language Processing

By Mark Hasegawa-Johnson, 3/2020



Watercolor, Antoine de Saint-Exupery, 1944. Public domain (copyright expired).

Photo, NYT Online, 1933. Public domain (copyright expired).

Outline

- Information Extraction
- Semantics
  - predicates, entities, and propositions
- Syntax
  - context-free grammar
  - parts of speech
- Training parsers: the Penn Treebank

Information Extraction

The prince loves his rose.

- Who?
- What?
- Where?
- When?
- Why?

Information Extraction

The prince loves his rose.

- Who? The prince and the rose.
- What? Love.
- Where? Not specified.
- When? Not specified.
- Why? Not specified.

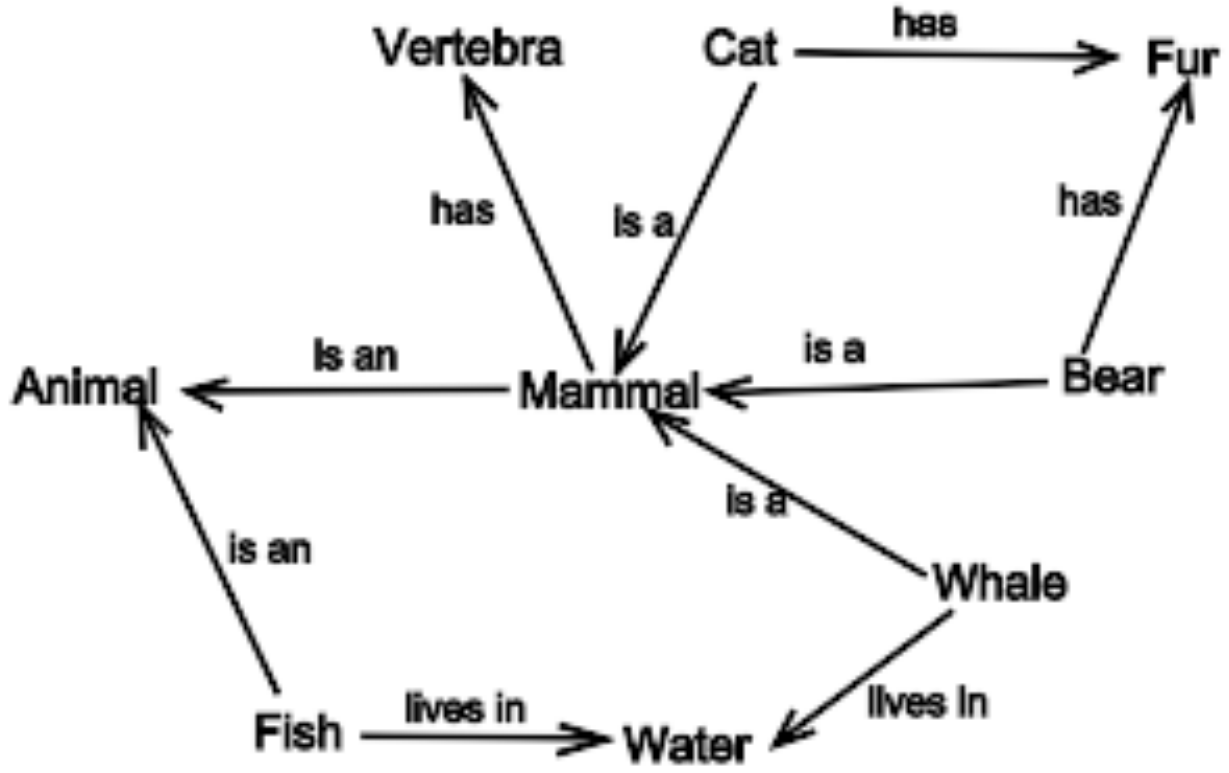
Information Extraction

Information Extraction (IE) seeks to

- Read free-form texts, like “the prince loves his rose.”
- Extract facts in a form that can be used for automatic reasoning processes.

Example: Semantic Network

- One possible target of IE is a semantic network, like the one shown at right
- The goal is to learn typed edges (relationships) connecting nodes (entities)



Example: RDF

Resource Description Framework (RDF) is a series of standards of the WWW consortium, in which facts are represented as triples:

- Subject
- Object
- Predicate

By W3C - http://www.w3.org/TR/rdf-primer/fig1dec16.png, Attribution, https://commons.wikimedia.org/w/index.php?curid=6218192

Outline

- Information Extraction
- Semantics
  - predicates, entities, and propositions
- Syntax
  - context-free grammar
  - parts of speech
- Training parsers: the Penn Treebank

Semantics

- Semantics = The study of natural language meanings
- Semantics considers that natural language is fundamentally about two types of things:
  - **Entities**: objects, people, animals, ideas, places, organizations…
  - **Predicates**: attributes, relationships, actions
- By applying a predicate to a set of entities, we get a **proposition**.
- The proposition is, itself, an idea, which can therefore, itself, be treated as an entity:

Antoine said that the prince loves his rose.

Predicates, Entities, and Propositions

- A **predicate** is like a function, that can be applied to some arguments.

Love(x,y)

- An **entity** is like a constant, which can be the value of the argument of a function:

Love("The prince", "his rose")

- When the arguments have been evaluated to constants, we get a **proposition**. For example, an RDF triple is a proposition with only two arguments: Subject and Object.

The Syntax-Semantics Interface

Suppose we are given the text

The prince loves his rose.

How can we get the propositional form

Love(the prince, his rose)

This problem is called the “syntax-semantics” interface, because one way to solve it is as follows:

1. Parse the sentence into noun phrases and verb phrases.
2. Map each noun phrase to an entity, each verb phrase to a predicate.

Outline

- Information Extraction
- Semantics
  - predicates, entities, and propositions
- Syntax
  - context-free grammar
  - parts of speech
- Training parsers: the Penn Treebank

Syntax

Syntax is the study of how words, in natural language, are grouped together in order to form meaningful phrases.

$$\big((the\ prince)_{NP}(loves\ (his\ rose)_{NP})_{VP}\big)_{S}$$

The structural elements are things like:

- NP = **noun phrase**, usually contains exactly one entity
- S = **sentence**, usually contains one proposition
- VP = **verb phrase**, usually contains a predicate with all of its arguments mapped to constants except the subject

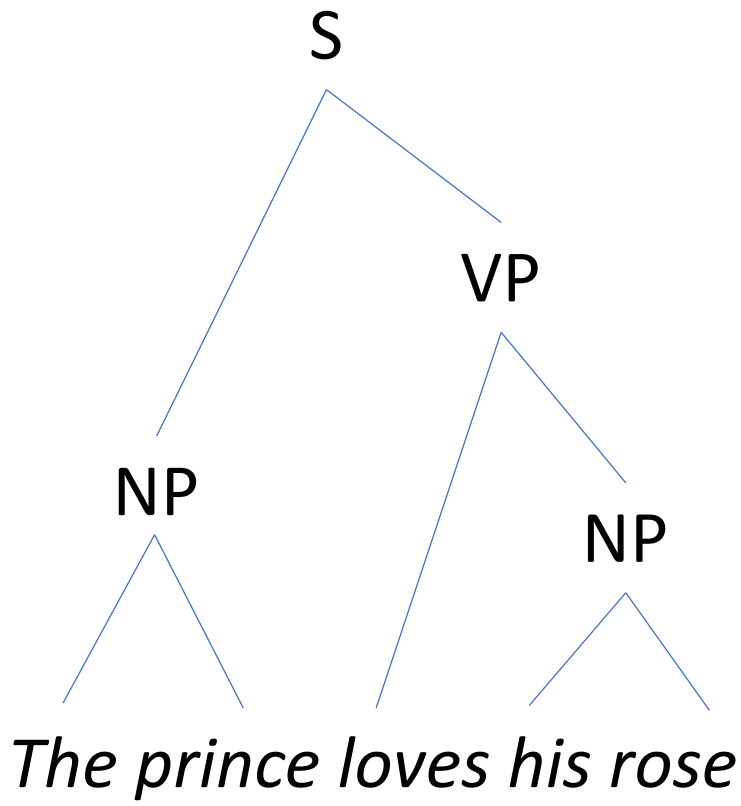
The structural elements of syntax *usually* correspond to meaning elements (entities, predicates, propositions) in semantics, but they don't have to.

Context-Free Grammar (CFG)

A “context-free grammar” (CFG) is a set of rules that specify how to produce a sentence.

A CFG is defined by three things:

1. A set of nonterminal nodes, $\mathcal{N}$ (structural elements)
2. A set of terminal nodes, $\mathcal{T}$ (words)
3. A set of production rules



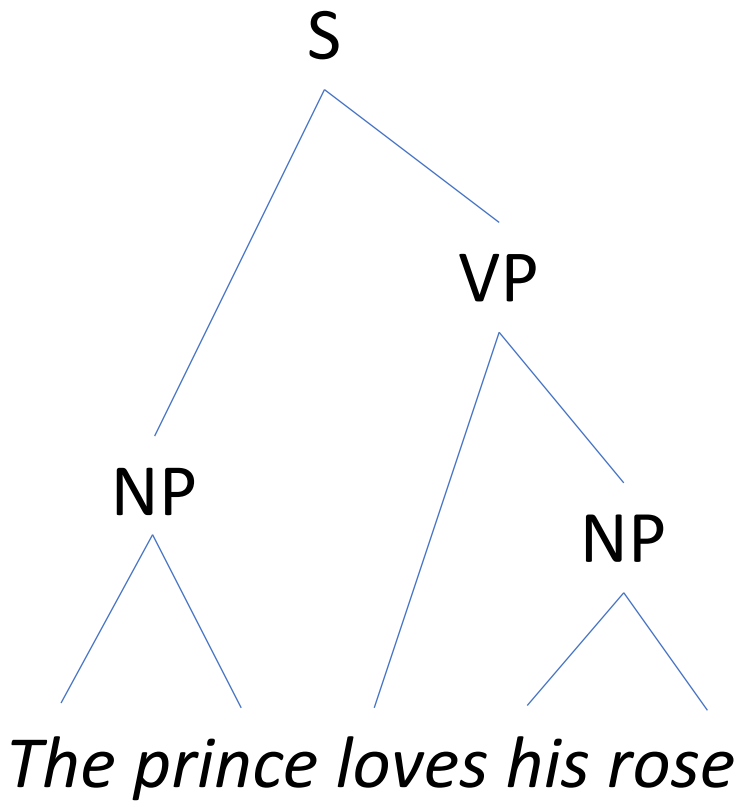
Context-Free Grammar (CFG)

Production rules always have the following form:

$$n \rightarrow (m|t) +$$

Where

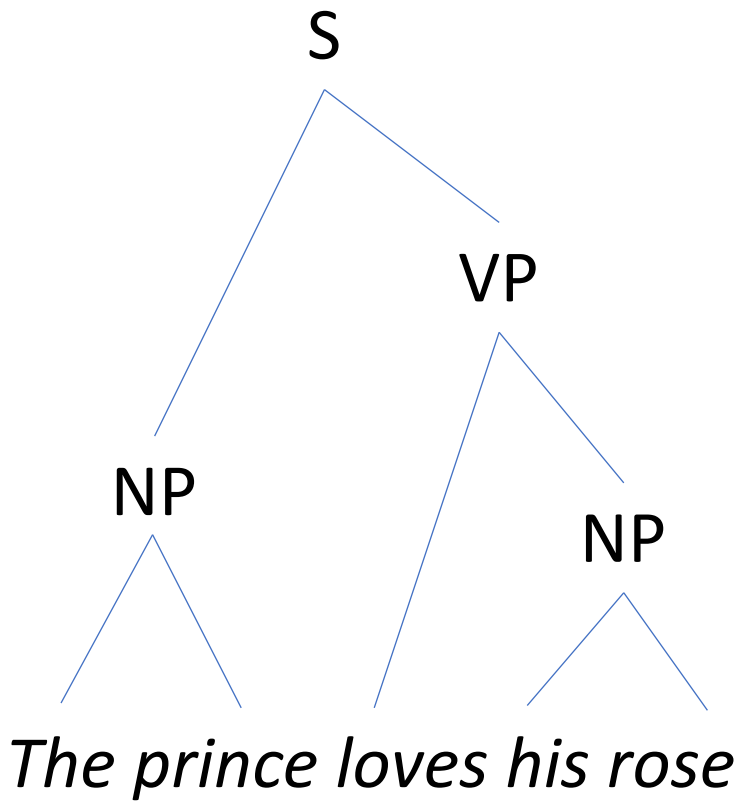
- $n, m \in \mathcal{N}$ are non-terminal nodes
- $t \in \mathcal{T}$ is a terminal node
- $(a|b)$ means “a OR b”
- $a +$ means “one or more examples of a”.



Context-Free Grammar (CFG)

So, to generate the sentence shown at right, we might use the following CFG:

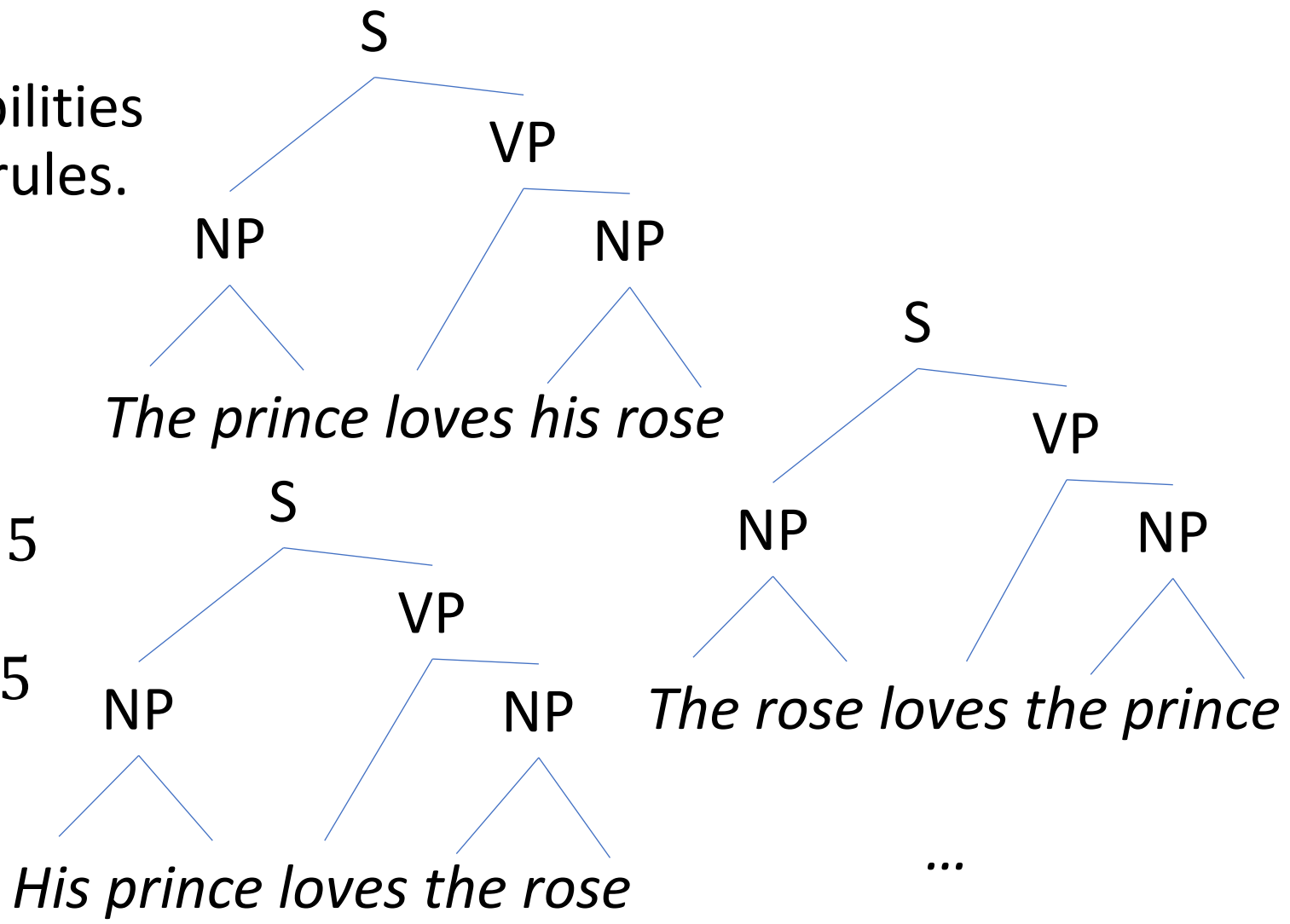
$S \rightarrow NP, VP$
$VP \rightarrow loves, NP$
$NP \rightarrow (the|his), (prince|rose)$



Probabilistic Context-Free Grammar (PCFG)

A PCFG is a CFG with probabilities attached to the production rules.
For example:

$P(S \rightarrow NP, VP) = 1$
$P(VP \rightarrow loves, NP) = 1$
$P(NP \rightarrow the\ prince) = 0.25$
$P(NP \rightarrow the\ rose) = 0.25$
$P(NP \rightarrow his\ prince) = 0.25$
$P(NP \rightarrow his\ rose) = 0.25$

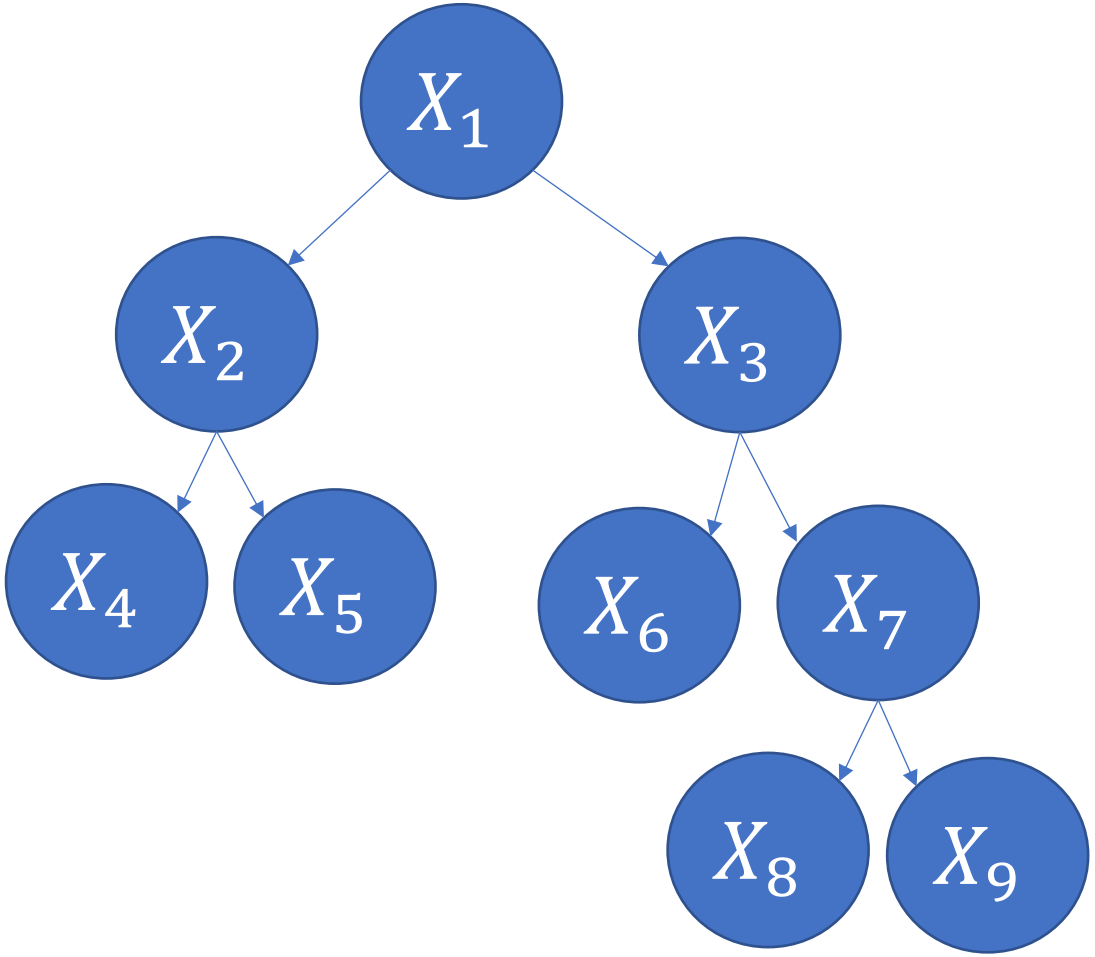


Probabilistic Context-Free Grammar (PCFG)

A PCFG can be written as a Bayes net!

- Every Nonterminal has 2 children.

Child1	Child2	Parent	P(C|P)
NP		S	1.0
	VP	S	1.0
the		NP	0.5
his		NP	0.5
	prince	NP	0.5
	rose	NP	0.5
…	…	…	…



Outline

- Information Extraction
- Semantics
  - predicates, entities, and propositions
- Syntax
  - context-free grammar
  - parts of speech
- Training parsers: the Penn Treebank

Parts of Speech

- At the bottom of each CFG, it has to generate a series of words.
- Listing all of the different words that can go in each slot is both tedious and unnecessary.
- It's usually much better to specify "parts of speech." For example:
  - DT (Determiner): a word like "a" or "the," specifies whether the entity named by a noun phrase is determinate or indeterminate.
  - JJ (Adjective): a word like "yellow," "big," "his," or "windy." Describes particular attributes of the entity.
  - NN (Noun): specifies the particular type of the entity.
- In this way, the production rules can be written in a more general form, e.g.,

$$NP \rightarrow (DT|JJ)\ NN$$

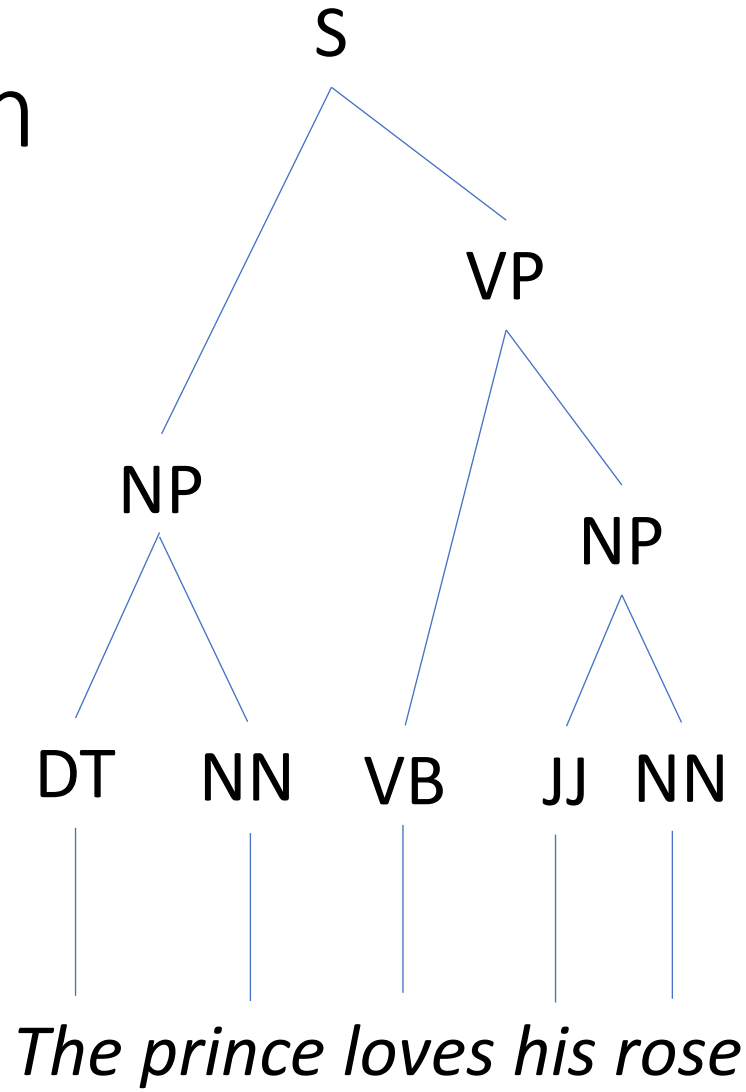
Normal-Form CFG

A Normal-Form CFG has just two types of rules:

1. $n_1 \rightarrow n_2\ n_3$, where n_1, n_2, and n_3 are all Nonterminals.
2. $n \rightarrow t$, where n is a Nonterminal and t is a Terminal (a word).

Example PCFG in Normal Form

Rule	Probability
$S \rightarrow NP\ VP$	1
$VP \rightarrow VB\ NP$	1
$NP \rightarrow DT\ NN$	0.5
$NP \rightarrow ADJ\ NN$	0.5
$VB \rightarrow loves$	1
$DT \rightarrow the$	1
$JJ \rightarrow his$	1
$NN \rightarrow prince$	0.5
$NN \rightarrow rose$	0.5



How to use PCFGs in practice

1. POS TAGGING: Figure out the part of speech of every word in the sentence.
2. PARSING: Use a Bayes net or some other parsing algorithm in order to figure out the structure of the sentence, i.e., which words belong to the first NP, which belong to the second NP.
3. INFORMATION EXTRACTION: Map each NP to an entity, and each VP to a predicate. Put the corresponding fact triple into your semantic net.

Outline

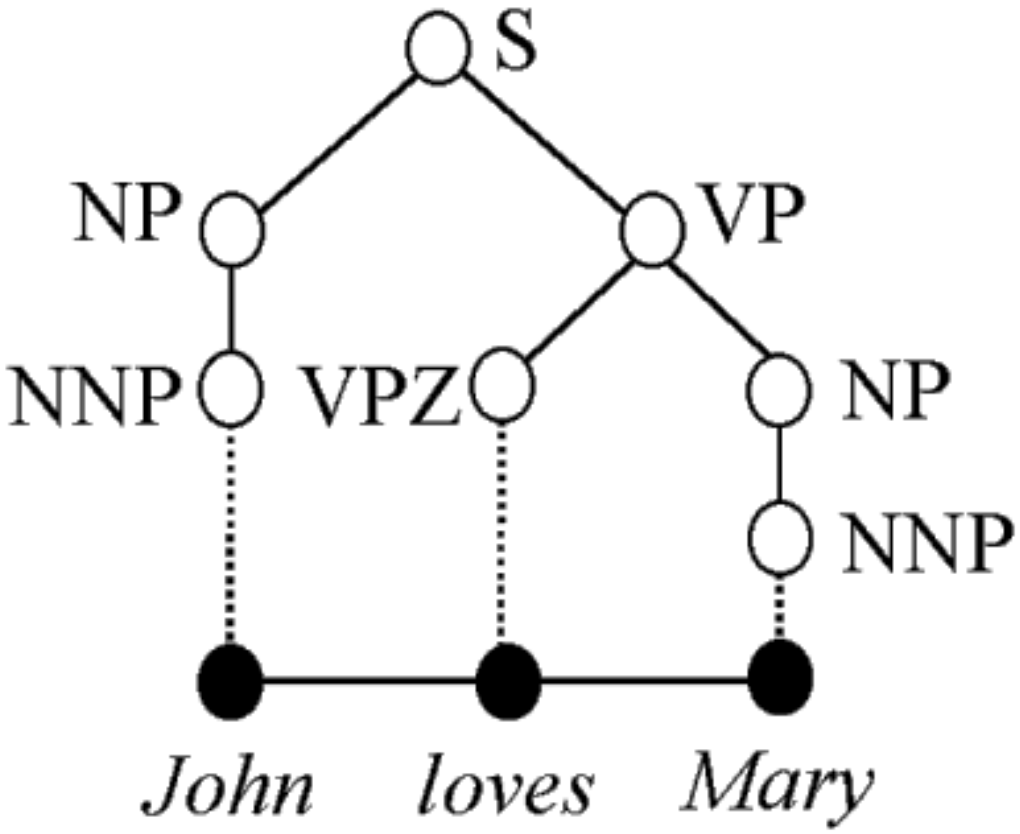
- Information Extraction
- Semantics
  - predicates, entities, and propositions
- Syntax
  - context-free grammar
  - parts of speech
- Training parsers: the Penn Treebank

Data: The Penn Treebank

- A “Treebank” is a database of
  - naturally occurring sentences,
  - marked up by linguists to show the phrase structure, using Nonterminal symbols (like NP and VP) and parts of speech (like NN, JJ, DT, and VB).
- Algorithms can then be trained to accept word strings as input, and generate parse structure as output.
- The Penn Treebank was the first large-scale publicly available treebank, created at the University of Pennsylvania in the early 1990s, using sentences from the Wall Street Journal.

Data: The Penn Treebank

```
(S (NP (NNP John))
    (VP (VPZ loves)
        (NP (NNP Mary)))
    (. .))
```



The Penn Treebank Parts of Speech

(Complete Listing, from https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html)

- **Verbs**: VB (verb), VBD (past), VBG (gerund), VBN (past participle), VBP (non-3rd person singular present), VBZ (3rd person singular present), MD (modal)
- **Nouns and pronouns**: NN (noun), NNS (plural), NNP (proper), NNPS (proper plural), PRP (personal pronoun), FW (foreign word), WP (wh-pronoun)
- **Adjectives and determiners**: JJ (adjective), JJR (comparative), JJS (superlative), CD (cardinal number), DT (determiner), PDT (predeterminer), POS (possessive ending), PRP$ (possessive pronoun), WDT (wh-determiner), WP$ (possessive wh-pronoun)
- **Adverbs**: RB (adverb), RBR (comparative), RBS (superlative), RP (particle), UH (interjection), WRB (wh-adverb)
- **Conjunctions & prepositions**: CC (coordinating conjunction), EX (existential there), IN (preposition or subordinating conjunction), LS (list item marker), TO (*to*)

What type of algorithms are trained using the Penn Treebank?

- Bayes nets or similar types of probabilistic algorithms
- Input: word strings
- Output:
  - For each word, what is its POS (part of speech)
  - For each POS, which phrase generated it
  - For each phrase, which phrase or sentence generated it … all the way back to the S (sentence) node, which governs the whole sentence

Next time: POS ambiguity

Don't desert me in the desert!

$VB \rightarrow desert$

…but also…

$NN \rightarrow desert$

By Nepenthes - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=5623273

How do we decide? Answer: a Bayes net called a "hidden Markov model" (HMM).