NLP 202: Syntax

Jeffrey Flanigan

Fall 2023

University of California Santa Cruz
jmflanig@ucsc.edu

Plan

- Background: Regular languages
- Pumping lemma
- Context free grammar (CFG)
- Phrase structure trees

Recall: Formal languages

- **Formal languages** are (usually infinite) sets of strings described mathematically
- Formal languages may be used to describe things that are not natural languages, e.g. computer languages

Linguistic Concept: Grammar

Native speakers have a **grammar**, a (infinite) set of sentences that they accept as being “well-formed”

a sentence can either be in the speaker’s grammar, or not in the speaker’s grammar

Example:

- * They says “hello.” **Linguists use * to indicate ungrammatical.**
- They say “hello.”

Recall: Regular languages

- A language $L \subset \Sigma^*$ is **regular** if there exists an FSA M such that $L(M) = L$ (DFA or NFA, they are equivalent)

Examples of finite-state technologies we have encountered

- n-gram language models are WFSAs
  The state is the previous $n-1$ words, weights are conditional probabilities
- HMMs are WFSTs
  The state is the previous tag, weights are emission and transition probabilities
- For the tag sequence, HMMs and CRFs are WFSAs
  The state is the previous tag, weights are transition probabilities

Examples of finite-state technologies we have encountered

- RNNs
  - Saturated RNNs (RNNs where the weights are very large, saturating the activation functions) can be proven to converge to WFSAs. Some evidence that real-world models are saturated
  - RNNs implemented on a computer are WFSAs, with a large state space - the state vector is a collection of n bits, so the number of states is 2^n
  - However, infinite precision RNNs are Turing complete

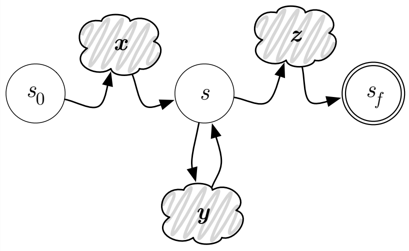
Are there languages that are not regular?

Yes.

Next: Languages that are not regular

Proving a Language Isn't Regular

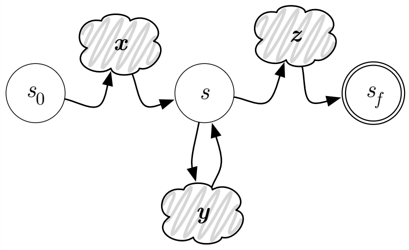
Pumping lemma (for regular languages): if L is an infinite regular language, then there exist strings $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$, with $\boldsymbol{y} \neq \epsilon$, such that $\boldsymbol{x}\boldsymbol{y}^n\boldsymbol{z} \in L$, for all $n \geq 0$.



Proof sketch: If FSA has m states, then for input of length $> m$ that is accepted, there must be a state s that we revisited. Therefore, we can keep revisiting that state.

Proving a Language Isn't Regular

Pumping lemma (for regular languages): if L is an infinite regular language, then there exist strings $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$, with $\boldsymbol{y} \neq \epsilon$, such that $\boldsymbol{x}\boldsymbol{y}^n\boldsymbol{z} \in L$, for all $n \geq 0$.

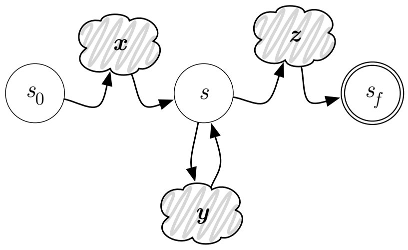


How to use:

If L is infinite and $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ do not exist, then L is not regular.

Proving a Language Isn't Regular

Pumping lemma (for regular languages): if L is an infinite regular language, then there exist strings $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$, with $\boldsymbol{y} \neq \epsilon$, such that $\boldsymbol{x}\boldsymbol{y}^n\boldsymbol{z} \in L$, for all $n \geq 0$.



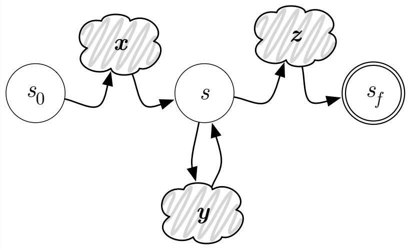
How to use:

If L is infinite and $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ do not exist, then L is not regular.

If L_1 and L_2 are regular, then $L_1 \cap L_2$ is regular.

Proving a Language Isn't Regular

Pumping lemma (for regular languages): if L is an infinite regular language, then there exist strings $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$, with $\boldsymbol{y} \neq \epsilon$, such that $\boldsymbol{x}\boldsymbol{y}^n\boldsymbol{z} \in L$, for all $n \geq 0$.



How to use:

If L is infinite and $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ do not exist, then L is not regular.

If L_1 and L_2 are regular, then $L_1 \cap L_2$ is regular.

If $L_1 \cap L_2$ is not regular, and L_1 is regular, then L_2 is not regular.

Example

$$L = \{a^n b^n | n \in \mathbf{N}\}$$
$$= \{ab, aabb, aaabbb, \ldots\}$$

L is infinite, but $\boldsymbol{x}$, $\boldsymbol{y} \neq \epsilon$, and $\boldsymbol{z}$ do not exist such that $\boldsymbol{x}\boldsymbol{y}^n\boldsymbol{z} \in L$, for all $n \geq 0$.
Therefore, L is not regular.

Is English a regular language?

Recursion

this is the house

this is the house that Jack built

this is the cat that lives in the house that Jack built

this is the dog that chased the cat that lives in the house that Jack built

this is the flea that bit the dog that chased the cat that lives in the house the Jack built

this is the virus that infected the flea that bit the dog that chased the cat that lives in the house that Jack built

Claim: English is not regular.

$$
\begin{aligned}
L_1 &= (\text{the cat}|\text{mouse}|\text{dog})^*(\text{ate}|\text{bit}|\text{chased})^* \text{ likes tuna fish} \\
L_2 &= \text{English} \\
L_1 \cap L_2 &= (\text{the cat}|\text{mouse}|\text{dog})^n(\text{ate}|\text{bit}|\text{chased})^{n-1} \text{ likes tuna fish}
\end{aligned}
$$

$L_1 \cap L_2$ is not regular, but L_1 is $\Rightarrow$ L_2 is not regular.

the cat likes tuna fish

the cat the dog chased likes tuna fish

the cat the dog the mouse scared chased likes tuna fish

the cat the dog the mouse the elephant squashed scared chased likes tuna fish

the cat the dog the mouse the elephant the flea bit squashed scared chased likes tuna fish

the cat the dog the mouse the elephant the flea the virus infected bit squashed scared chased likes tuna fish

Linguistic Debate

Chomsky put forward an argument like the one we just saw.

(Chomsky gets credit for formalizing a hierarchy of types of languages: regular, context-free, context-sensitive, recursively enumerable. This was an important contribution to CS!)

Some are unconvinced, because after a few center embeddings, the examples become unintelligible.

Nonetheless, most agree that natural language syntax isn't well captured by FSAs.

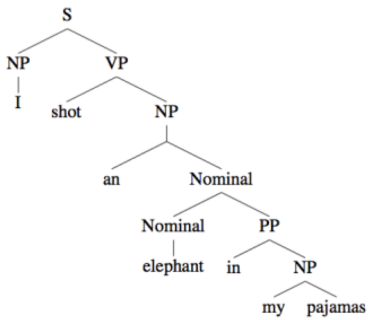
Context-free grammar (CFG)

It seems FSAs/regular languages may not have the representation power that we need for natural language.

Let's look at something more powerful: **Context-free grammar (CFG)**.

Syntax

- Syntax: from Greek syntaxis “setting out together, arrangement”
- Refers to the way words are arranged together, and the relationship between them
- Goal of syntax is to model the knowledge of that people unconsciously have about the grammar of their native language



PRP VBD DT NN IN PRP$ NNS

I shot an elephant in my pajamas

Why Is Syntax Important?

- Grammar checkers
- Question answering
- Information extraction
- Machine translation

Why Is Syntax Important?

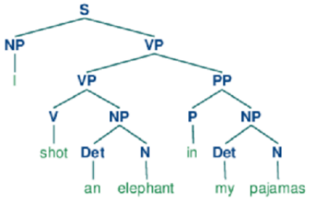
Linguistic typology; relative positions of subjects (S), objects (O), and verbs (V)

SVO	English, Mandarin	I grabbed the chair
SOV	Latin, Japanese	I the chair grabbed
VSO	Hawaiian	Grabbed I the chair
OSV	Yoda	Patience you must have
…	…	…

Formalisms

Phrase structure grammar
(Chomsky 1957)

Dependency grammar
(Mel'čuk 1988; Tesnière 1959; Pāṇini)



Context-free grammar

N	Finite set of non-terminal symbols	NP, VP, S
Σ	Finite alphabet of terminal symbols	the, dog, a
R	Set of production rules, each $A \rightarrow \beta$ $\beta \in (\Sigma, N)$	S → NP VP Noun → dog
S	Start symbol	

Example

Production Rules

- A $\rightarrow$ Tim
- B $\rightarrow$ saw
- B $\rightarrow$ saw B
- A $\rightarrow$ Pat
- S $\rightarrow$ A B

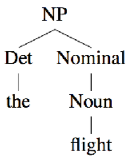
Start symbol S

Example derivation

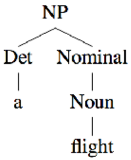
1. S
2. A B
3. A saw
4. Tim saw

Derivation

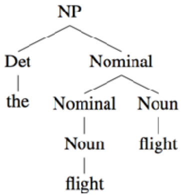
Given a CFG, a derivation is the sequence of productions used to generate a string of words (e.g., a sentence), often visualized as a parse tree.



the flight



a flight



the flight flight

Language

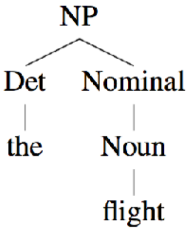
The formal language defined by a CFG is the set of strings derivable from S (start symbol)

Infinite strings with finite productions

- This is the house
- This is the house that Jack built
- This is the cat that lives in the house that Jack built
- This is the dog that chased the cat that lives in the house that Jack built
- This is the flea that bit the dog that chased the cat that lives in the house the Jack built
- This is the virus that infected the flea that bit the dog that chased the cat that lives in the house that Jack built

Smith 2017

Bracketed notation



$[_{\text{NP}}\ [_{\text{Det}}\ \text{the}]\ [_{\text{Nominal}}\ [_{\text{Noun}}\ \text{flight}]]]$

Context free languages

A language that can be described with a CFG, is called a **context free language**.

All regular languages are context free languages

(conversion process: have non-terminal for each state, and a production rule for each edge).

Context-free languages

A language that can be described with a CFG, is called a **context-free language**.

All regular languages are context-free languages.

Our non-regular language

$$\begin{aligned} L &= \{a^n b^n | n \in \mathbf{N}\} \\ &= \{ab, aabb, aaabbb, \ldots\} \end{aligned}$$

Is this context free?

Our non-regular language

$$L = \{a^n b^n | n \in \mathbf{N}\}$$
$$= \{ab, aabb, aaabbb, \ldots\}$$

Yes

- $N = \{X\}$
- $\Sigma = \{a, b\}$
- $R = \{X \to aXb, S \to ab\}$
- $S = X$

An Example CFG for a Tiny Bit of English

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP

Det → that | this | a
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Houston | NWA
Aux → does
Preposition → from | to | on | near
| through

(Phrase-Structure) Recognition and Parsing

Given a CFG $(\mathcal{N}, S, \Sigma, \mathcal{R})$ and a sentence $\boldsymbol{x}$, the **recognition** problem is:

Is $\boldsymbol{x}$ in the language of the CFG?

Related problem: **parsing**

Show one or more derivations for $\boldsymbol{x}$, using $\mathcal{R}$.

These are related: the proof is a derivation.

In general, in NLP, parsing is the production of a structure (tree, graph, etc) from an input sentence.

Beyond Context-Free: The Chomsky hierarchy

Grammar	Languages	Automaton	Production rules (constraints)*	Examples[3]
Type-0	Recursively enumerable	Turing machine	$\gamma \rightarrow \alpha$ (no constraints)	$L = \{w \mid w$ describes a terminating Turing machine$\}$
Type-1	Context-sensitive	Linear-bounded non-deterministic Turing machine	$\alpha A \beta \rightarrow \alpha \gamma \beta$	$L = \{a^n b^n c^n \mid n > 0\}$
Type-2	Context-free	Non-deterministic pushdown automaton	$A \rightarrow \alpha$	$L = \{a^n b^n \mid n > 0\}$
Type-3	Regular	Finite state automaton	$A \rightarrow \mathrm{a}$ and $A \rightarrow \mathrm{a}B$	$L = \{a^n \mid n \geq 0\}$

* Meaning of symbols:

- a = terminal
- A, B = non-terminal
- α, β, γ = string of terminals and/or non-terminals
    - α, β = maybe empty
    - γ = never empty

Context-Sensitive Example

Production Rules

- A $\rightarrow$ Tim
- B $\rightarrow$ saw
- A $\rightarrow$ Pat
- S $\rightarrow$ A B
- A B $\rightarrow$ Tim **This is a context-sensitive rule (not allowed in CFGs)**

Start symbol S

Example derivation

1. S
2. A B
3. Tim

Context-sensitive grammars used in NLP

- Lexical functional grammar (LFG)
- Combinatory categorial grammar (CCG)
- Head-driven phrase structure grammar (HPSG)
- Tree-adjoining grammar (TAG)

So far we've been talking about formal languages.

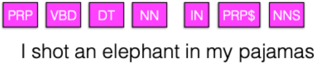
Let's talk about the structure of natural language:
the linguistic notion of **syntax**.

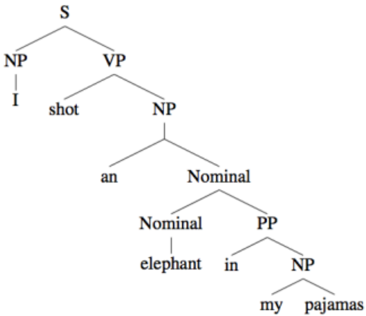
Linguistics and syntax

Phrase-structure trees

Syntax

- Syntax: from Greek syntaxis “setting out together, arrangement”
- Refers to the way words are arranged together, and the relationship between them
- Goal of syntax is to model the knowledge of that people unconsciously have about the grammar of their native language





PRP VBD DT NN IN PRP$ NNS

I shot an elephant in my pajamas

Constituency

- Groups of words (“constituents”) behave as single units
- “Behave” = show up in the same distributional environments

How words group into units and how the various kinds of units behave ?

Parts of speech

- Parts of speech are categories of words defined distributionally by the morphological and syntactic contexts a word appears in.

from POS 2/20

Syntactic distribution

- Substitution test: if a word is replaced by another word, does the sentence remain grammatical?

Kim saw the	elephant	before we did
	dog	
	idea	
	*of	
	*goes	

Bender 2013

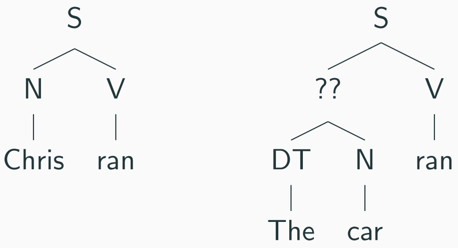
from POS 2/20

Language is Hierarchical

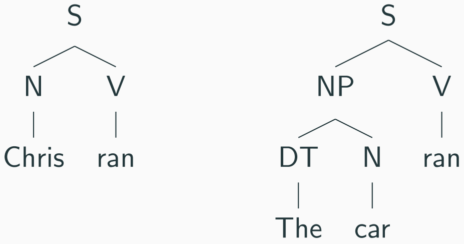
- The car is fast.
- The truck is fast.
- The loud truck is fast.
- The very loud truck is fast.
- The very, very loud truck is fast.
- The very, very, very loud truck is fast.

We can replace nouns with other nouns, and with other phrases that behave like nouns. These are noun phrases.

Nouns



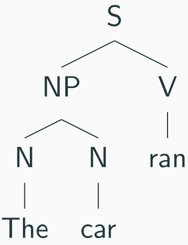
Nouns

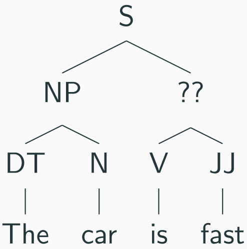


Noun Phrases (NPs)

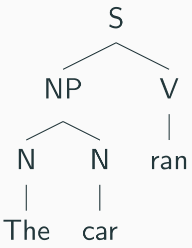
- The elephant arrived.
- It arrived.
- Elephants arrived.
- The big ugly elephant arrived.
- The elephant I love to hate arrived.

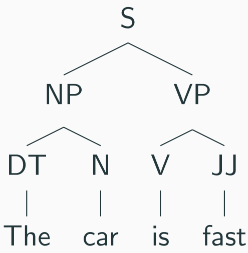
Verbs



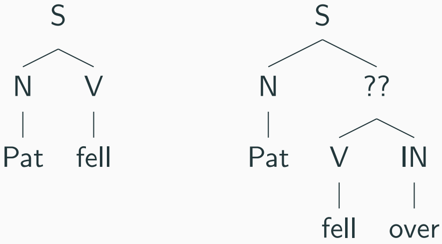


Verbs

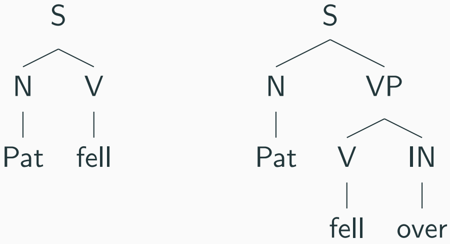




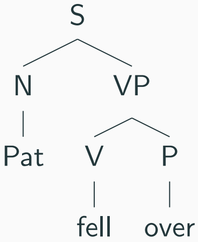
Prepositions

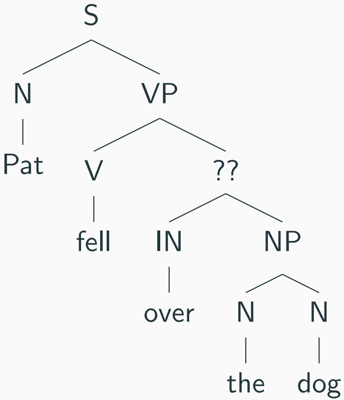


Prepositions

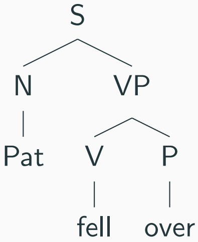


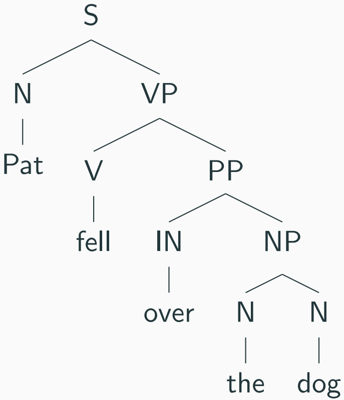
Prepositions





Prepositions





Prepositional Phrases (PPs)

- I arrived on Tuesday.
- I arrived in March.
- I arrived under the leaking roof.

Every **prepositional phrase** contains a **noun phrase**.

Sentences or Clauses (Ss)

- John loves Mary.
- John loves the woman he thinks is Mary.
- Sometimes, John thinks he is Mary.
- It is patently false that sometimes John thinks he is Mary.

Clauses

Sentences consist of one or more clauses. Each clause has a verb, and looks like a mini-sentence.

- I saw Pat.
- Alex thinks that [I saw Pat]. **Embedded clause**
- I found a truck.
- She found a truck.
- She found [the truck that [I found]]. **Noun-phrase with a relative clause**

Constituents

Constituents are groups of words that "go together."

Linguists characterize constituents in a number of ways, including:

Constituents

Constituents are groups of words that "go together."

Linguists characterize constituents in a number of ways, including:

- where they occur (e.g., "NPs can occur before verbs")
- where they can *move* in variations of a sentence
    - On September 17th, I'd like to fly from Atlanta to Denver
    - I'd like to fly on September 17th from Atlanta to Denver
    - I'd like to fly from Atlanta to Denver on September 17th

Constituents

Constituents are groups of words that "go together."

Linguists characterize constituents in a number of ways, including:

- where they occur (e.g., "NPs can occur before verbs")
- where they can *move* in variations of a sentence
  - On September 17th, I'd like to fly from Atlanta to Denver
  - I'd like to fly on September 17th from Atlanta to Denver
  - I'd like to fly from Atlanta to Denver on September 17th
- what parts can move and what parts can't
  - *On September I'd like to fly 17th from Atlanta to Denver

Constituents

Constituents are groups of words that "go together."

Linguists characterize constituents in a number of ways, including:

- where they occur (e.g., "NPs can occur before verbs")
- where they can *move* in variations of a sentence
  - On September 17th, I'd like to fly from Atlanta to Denver
  - I'd like to fly on September 17th from Atlanta to Denver
  - I'd like to fly from Atlanta to Denver on September 17th
- what parts can move and what parts can't
  - *On September I'd like to fly 17th from Atlanta to Denver
- what they can be conjoined with
  - I'd like to fly from Atlanta to Denver on September 17th and in the morning

Example

- Sam climbed up the ladder.
- Sam picked up the ladder.

Tests for constituency: Movement test

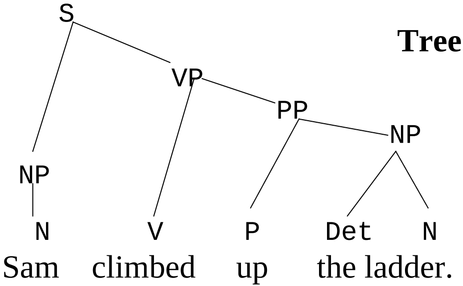
- Sam climbed up the ladder.
- Up the ladder Sam climbed.
- Sam picked up the ladder.
- *Up the ladder Sam picked.

In "Sam climbed [up the ladder]," "up the ladder" is a constituent, but not for "Sam picked up the ladder."

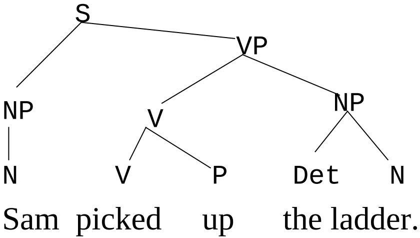
Tests for constituency: Coordination test

- Sam climbed up the ladder.
- Sam climbed out the window.
- Sam climbed up the ladder and out the window.
- Sam picked up the ladder.
- Sam picked out some shoes.
- *Sam picked up the ladder and out some shoes.

Tree 1


S
VP
PP
NP
NP
N
V
P
Det
N
Sam climbed up the ladder.

Tree 2


S
VP
NP
V
NP
N
V
P
Det
N
Sam picked up the ladder.

Annotating Phrase Structure

- These structures are called **phrase structure trees**
- You can use procedures like this create a set of conventions for annotation
- It can get tricky, and there are many borderline cases (convincing arguments for two different ways)
- Researchers decide a set of **annotation conventions**, written up in the **annotation guidelines**
- You can use these guidelines to annotate a **treebank**

Annotation conventions differ between treebanks, and they are all valid (with pros and cons for annotation decisions)