Lecture 7:
Grammatical Representation:
Syntax and a little Morphology

What is Grammar?

- Several definitions:
  - The rules and principles that guide the structure of sentences in language
  - Regularities in the mapping between phonological form and semantic form
  - The way a language speaker organizes her experience with language, allowing her to make generalizations

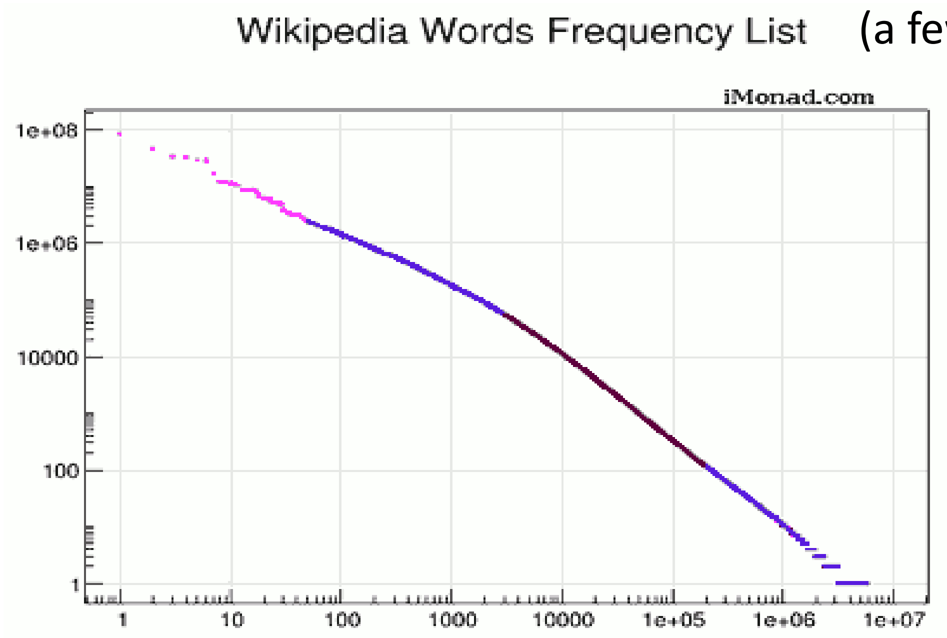
Why do we need grammar?

- Grammatical analysis in NLP involves breaking the text into simpler parts and categorizing them
  - New sentences are the rule, rather than the exception (unlike new words)

- Two main motivations for grammar:
  - Structural Analysis: decomposing complex units into simpler sub-structures can assist learning and generalization
  - Semantic Information: grammatical structure often reflects semantic structure and distinctions

The Importance of Categorization

- Reminder: linguistic phenomena tend to have a power-law distribution



(a few billions of words)

freq.

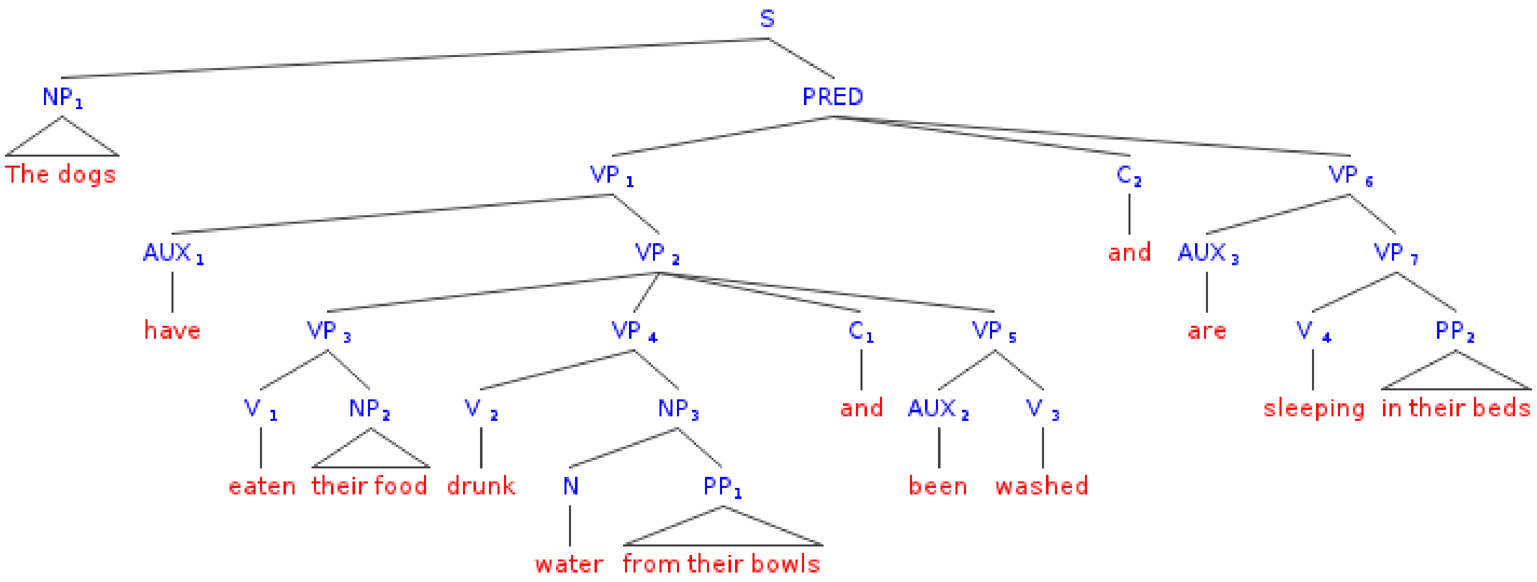
rank

$$\text{freq} \propto \frac{1}{rank^c}$$

- Most word types are rare
- Many word instances are rare

The Importance of Categorization

- This means that for most types, we don’t have meaningful statistics
  - Many types don’t appear at all
- Categorization allows information to propagate from one type to another



Overview of Grammar: Words

Basic Linguistic Theory, Vol. 1-2, Dixon (2010)

- **Word:** surprisingly difficult to define
  - **Practical:** anything that is between two spaces
  - But: what about spoken language?
  - what about phrases whose spaces disappeared? “kickstart” or “kick-start”?)
  - What about languages that don’t have spaces?
    (e.g., Mandarin Chinense)

Overview of Grammar: Words

- Sapir (1921): “one of the smallest, completely satisfying bits of isolated meaning into which the sentence resolves itself”

- Zirmunskij (1966): “the word is the most concise unit of language, which is independent in meaning and form”

- Bloomfield (1933): “a word, then, is a free form which does not consist entirely of (two or more) lesser free forms; in brief, a word is a *minimum free form*”
  - “free form” is interpreted to mean words that can occur on their own, with a given meaning
  - But: many words (“the”, “my”, “and”) cannot occur on their own

Overview of Grammar: Words

- We will distinguish between:
    - **Phonological words:** defined by phonological criteria, primarily: syllable structure, pause phenomena, prosodic features (stress, tone, nasalization, retroflexion etc.), phonological word boundary rules
    - **Grammatical words:** have a conventionalized meaning, can undergo morphological processes (e.g., affixation, reduplication, vowel shift)

Morphological Analysis

- Minimally: separating out clitics (*tokenization*)
  - John’s → John + ‘s
  - Auf’m → auf ‘m (German: auf+dem)
  - לבית הספר → ל+בית ה+ספר
- Base forms (lemmatization, omitting inflectional morphology)
  - takes → take
  - dogs → dog
  - יתנו → נתן (what about ינתנו?)
- Segmentation into morphemes:
  - renewal → re + new + al
  - nachgedacht → nach + ge__t + dach (denken)
  - מטבחונים → מטבח + ון + ים

Basic Syntactic Units: Clauses

- **Clause:** “the description of some activity, state or property”
- Simple clauses have a predicate, core arguments (usually subjects and objects), and non-core arguments (e.g., location, manner, instrument)
  - “John **kicked** the ball”
  - “The door **opened**”
  - “The door silently **opened**”
  - “John opened the door with a hammer”
  - “John is **tall**”

Inter-Clause Relations

- Three main inter-relations between clauses within a sentence: complement clauses, relative clauses, subordinate clauses, coordination
- Complement clauses:
  - Clauses that serve as an argument in another clause
    - "John said **he will kick the ball**"
    - "Mary convinced Paul **to join them**"
- Relative clauses:
  - Clauses that modify an argument of another clause
    - "John, **who has studied the language for years**, speaks German well"
    - "Eagles that **fly** swim"
- Linked clauses:
    - After **he had studied it for years**, **John could speak German well**
    - Although **he had studied it for years**, **John could speak German well**

Basic Units: Noun Phrases

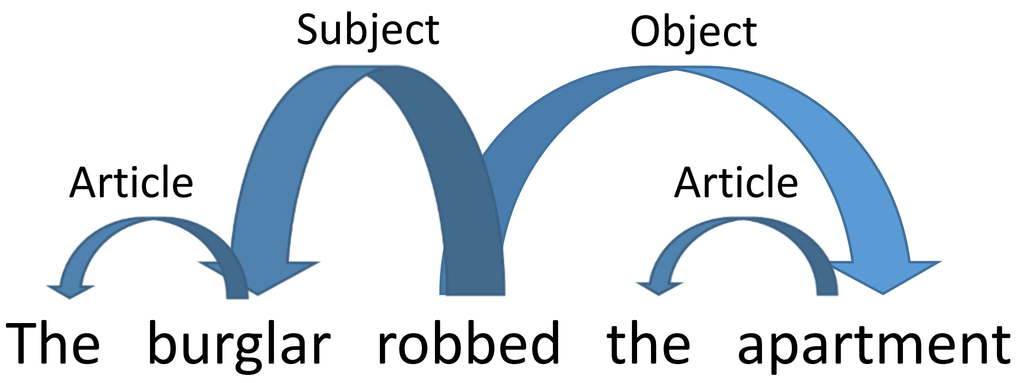
- **Noun phrase:** an argument of a clause is generally a noun phrase. Two major classes:
  - Proper nouns or names (of people, places, organizations etc.)
  - Common nouns, possibly along with some modifiers
    - Possible modifiers: adjectives (e.g., fast, thin), relative clauses, cardinals or ordinals (e.g., one, first), modifying nouns (e.g., shipping company)
  - Noun phrases generally have a *head*, a word that determines the sematic type of the whole phrase, and is an essential component of its meaning
    - Shipping **company**
    - Fast **runner**
    - The **man** who wasn’t there

Basic Units: Syntactic Heads

- The notion of a *head* is used in a wider context than noun phrases.
- The verb is considered the head of a clause:
  - Arguments and modifiers are considered its *dependents*
    - **John** ran **home** ("ran" is the head of the clause, "John" and "home" are its dependents)
    - **Bill stupidly** answered **the officer's question** ("answered" is the head, "Bill", "stupidly" and "the officer's question" are its dependents)
- Some schemes view all syntactic relations as head-dependent relations (*dependency schemes*)

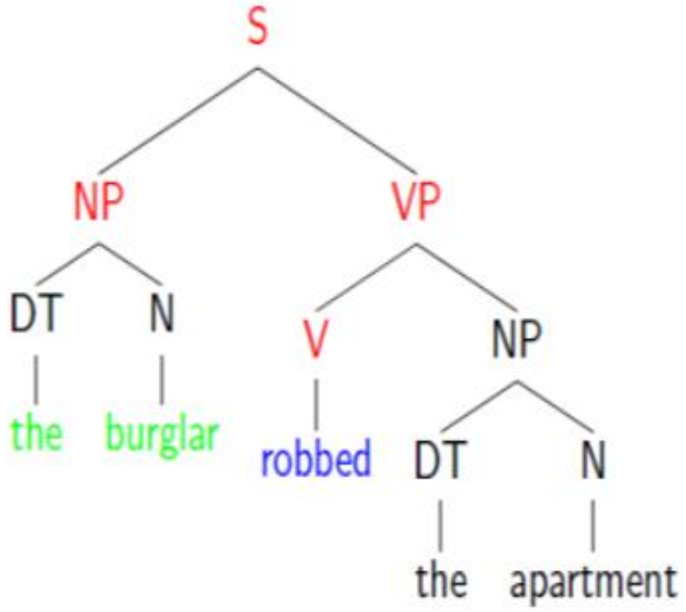
Two Common Representations Types

- ***Dependency parses***: syntax is represented as head-dependent relations between words
  - Nodes are words, edges are relations
  - The arguments and modifiers of a word are its children
  - Edge labels mark the type of relation
  - The verb of the main clause is the root



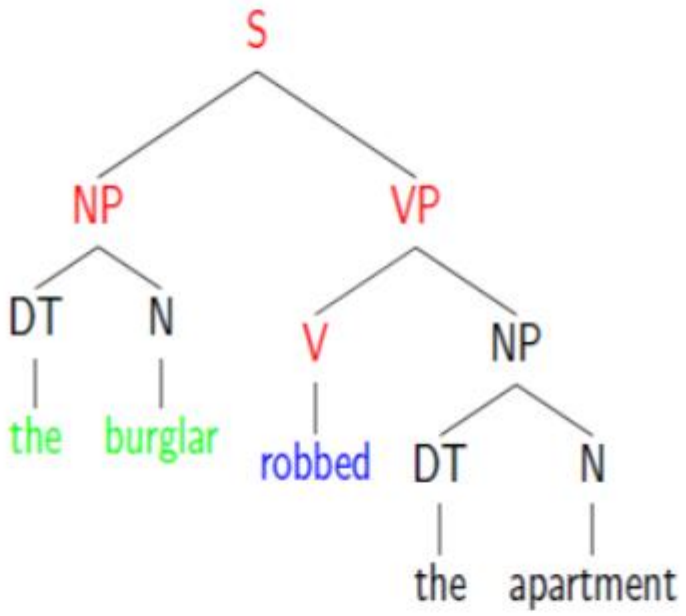
Two Common Representations Types

- ***Phrase-based (constituency) parses:*** represent the sentence as a collection of nested phrases
  - Words are at the leaves
  - Non-terminal labels represent the phrase type. This is determined by the type of the headword (e.g., Noun Phrases (NP) are phrases head by a noun)



Information from Syntactic Structure

1. Phrases
2. Grammatical relations
3. Syntactic disambiguation
4. A proxy to semantics



Information from Syntactic Structure

- Phrases:
  - Syntactic structure decomposes the sentence to sub-strings
  - These can be useful for applications

  - For example: in machine translation, it is sometimes possible to phrase translation rules as re-ordering of phrases, rather than of words

Machine Translation Example

- Compare English with Japanese glosses:

"IBM bought Lotus"

"IBM Lotus bought"

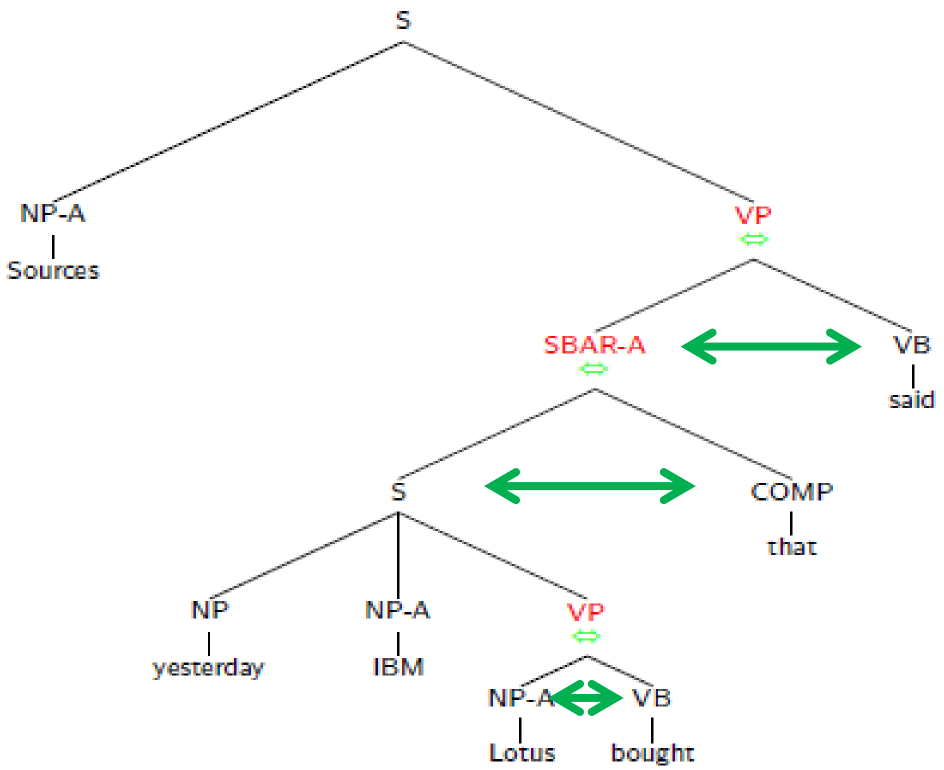
"Sources said that IBM bought Lotus yesterday"

"Sources yesterday IBM Lotus bought that said"

- It is difficult to phrase the possible permutations in terms of words, easier in terms of phrases

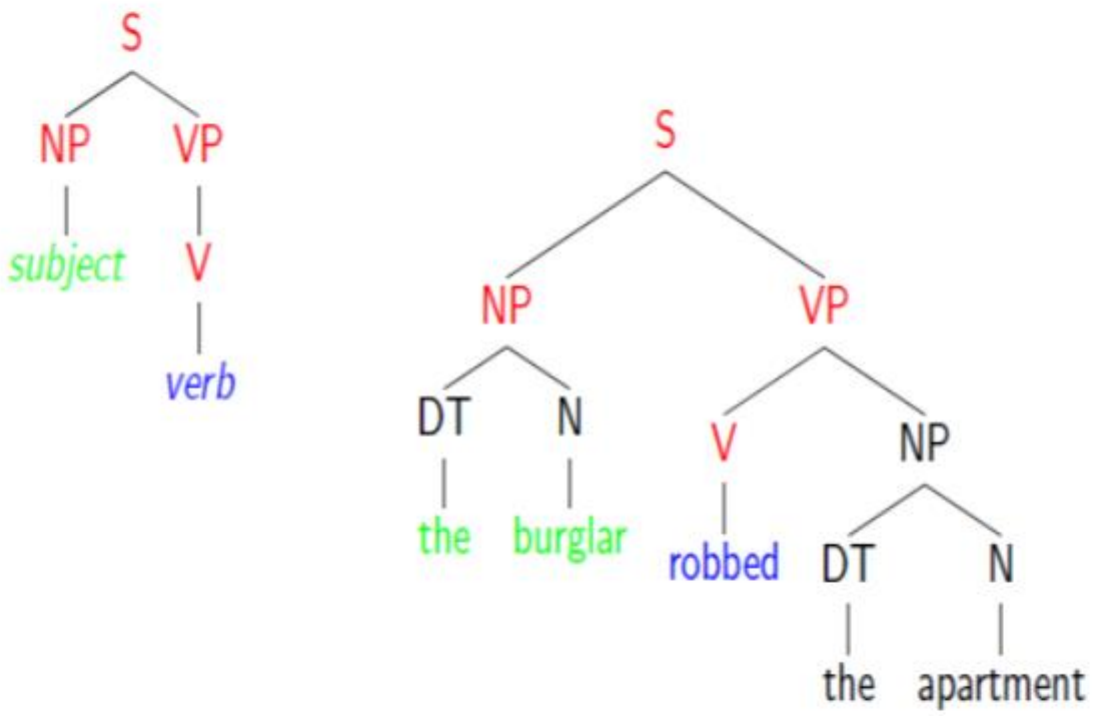
Machine Translation Example

- This can be seen by comparing English and Japanese Constituency Structures:



Information from Syntactic Structure

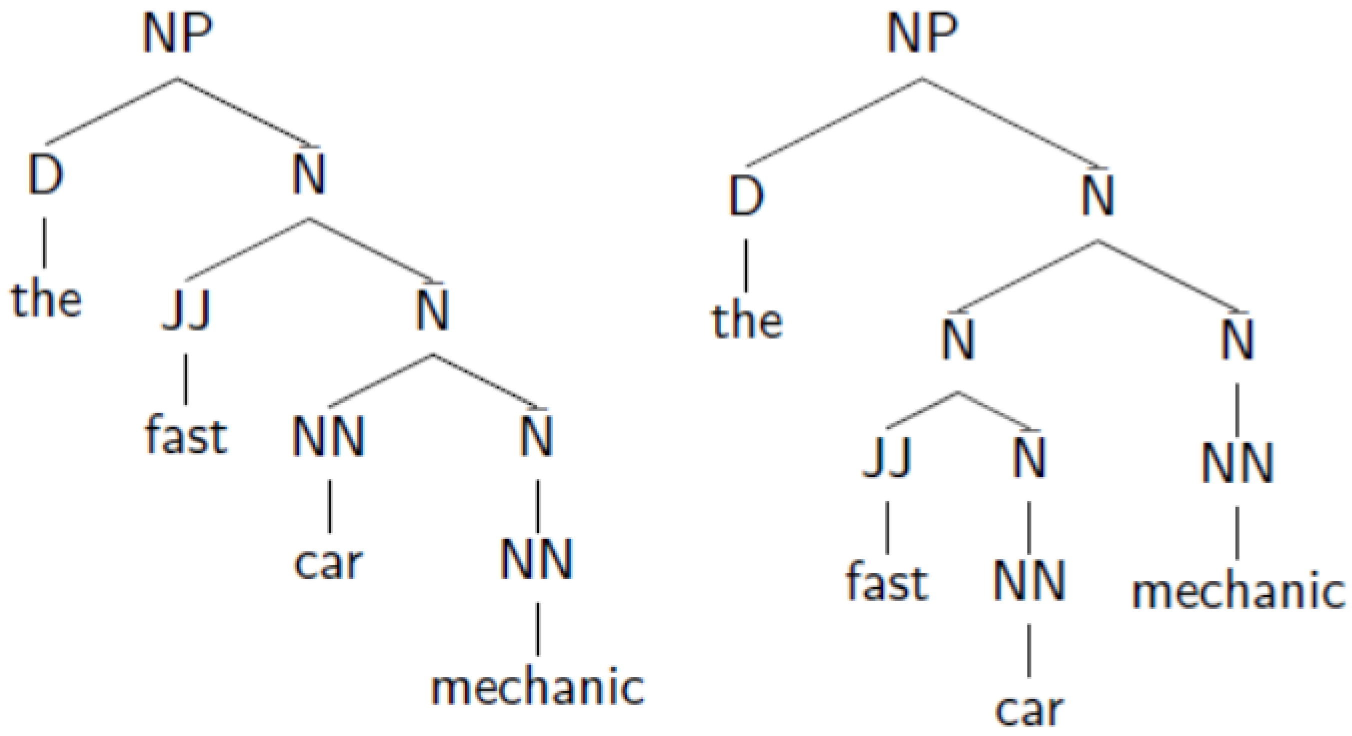
- Extracting grammatical relations:



- *the burglar* is the **subject** of robbed
- *the apartment* is the **object** of robbed

Information from Syntactic Structure

- Syntactic disambiguation: (adjective scope)



Examples of Syntactic Ambiguity

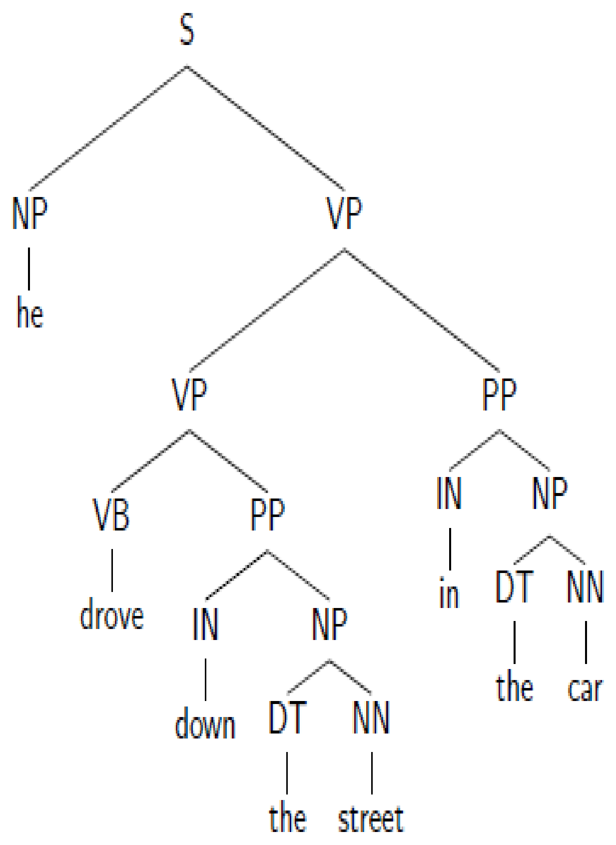
- Prepositional phrases:
  - *They cooked the beans in the pot on the stove with handles*
- Particle vs. preposition:
  - *The puppy tore up the staircase*
- Complement structures
  - *The tourists objected to the guide that they couldn’t hear*
  - *She knows you like the back of her hand*
- Gerund vs. participipial adjective
  - *Visiting relatives can be boring*
  - *Changing schedules frequently confused passengers*

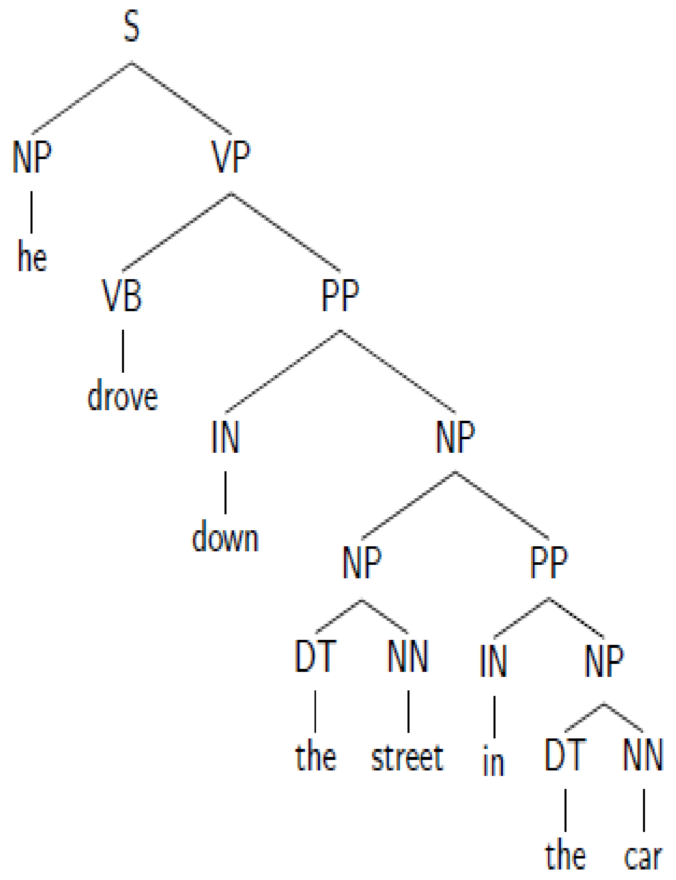
Examples of Syntactic Ambiguity

- Modifier scope within NPs
  - impractical design requirements
  - plastic cup holder
- Multiple gap constructions
  - The chicken is ready to eat.
  - The contractors are rich enough to sue.
- Coordination scope:
  - Small rats and mice can squeeze into holes or cracks in the wall.

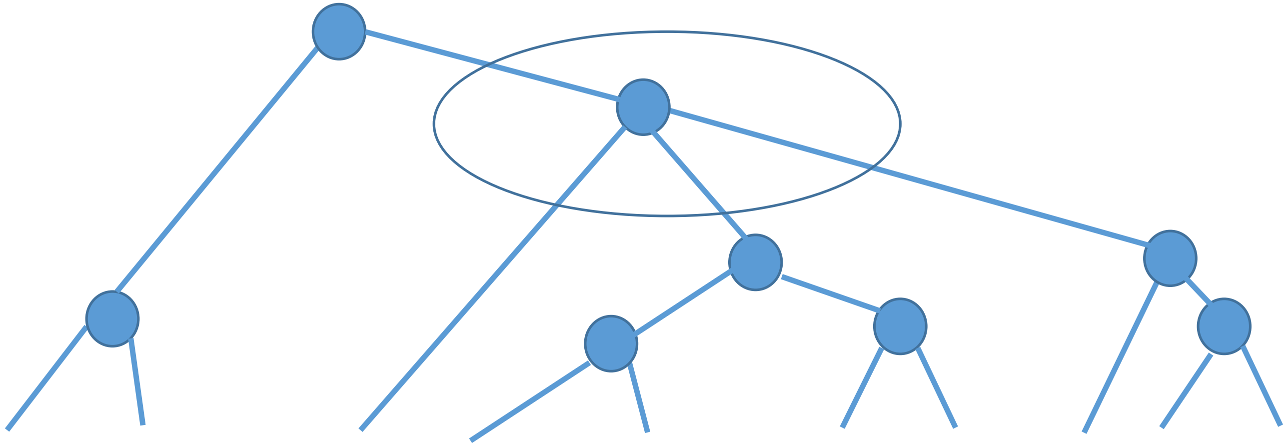
Information from Syntactic Structure

- Syntactic disambiguation: (prepositional phrase attachment)

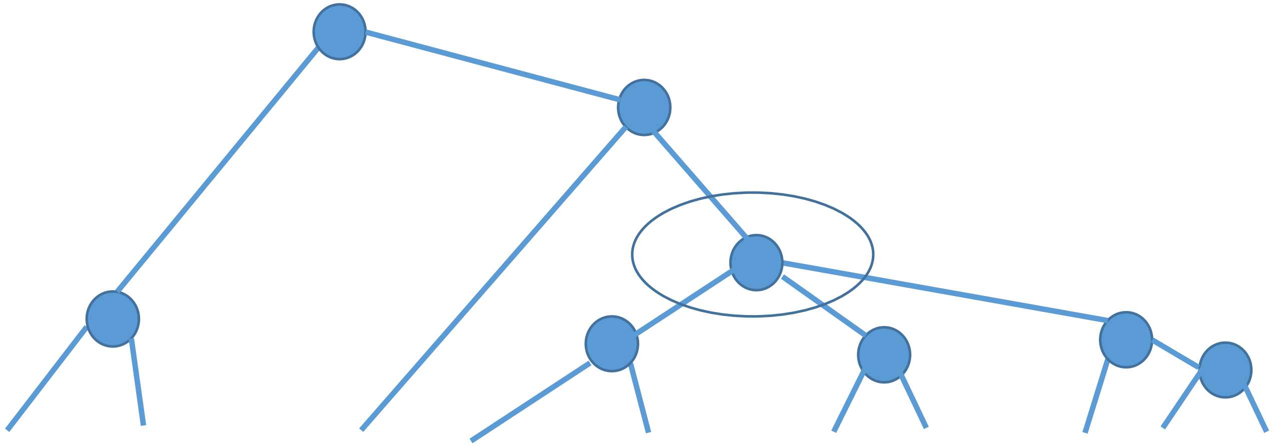




Information from Syntactic Structure



The terrorists took the weapons they found **to** the hangar



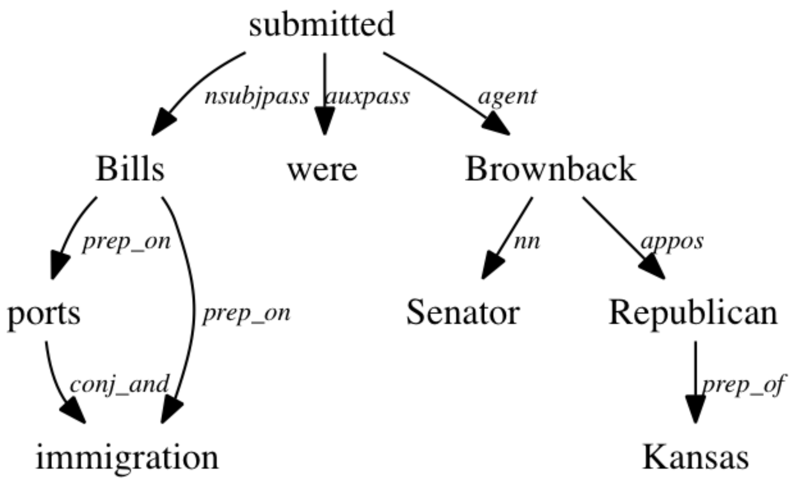
The terrorists took the weapons they found **in** the hangar

A Proxy to Semantics

- Syntactic structure provides valuable semantic information that can serve downstream applications, such as paraphrasing, machine translation, summarization etc.

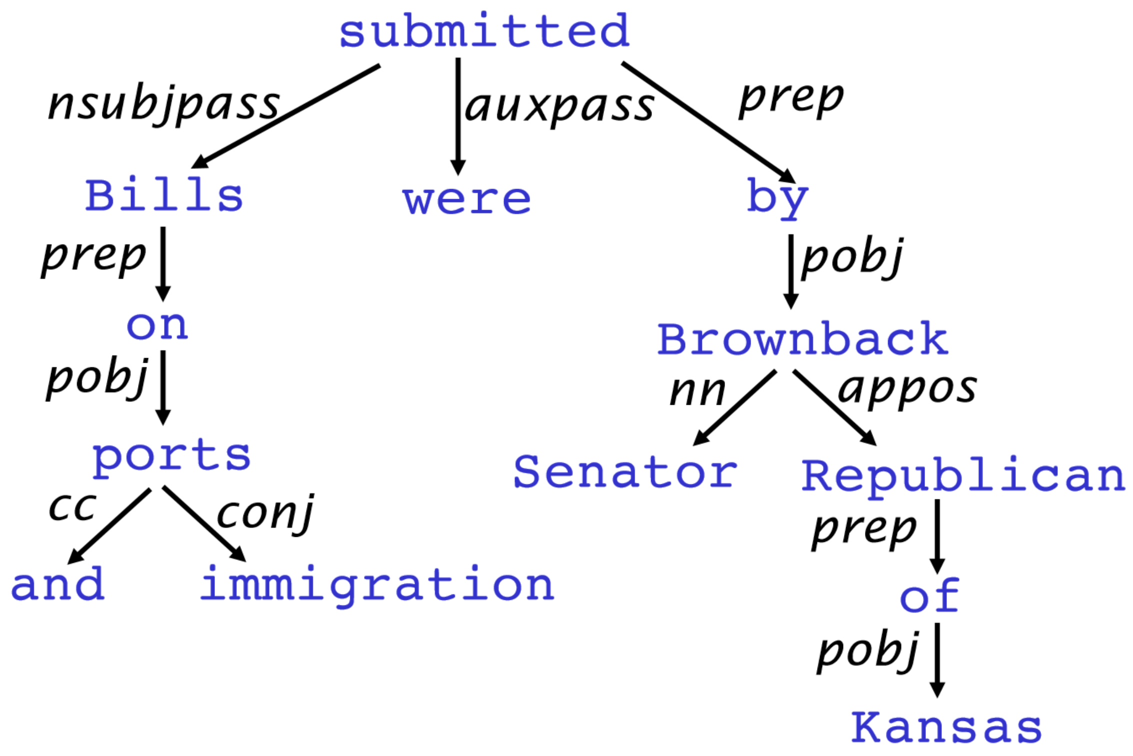
The event structure is clearer from here

Bills on ports and immigration were submitted by Senator Brownback, a republican from Kansas



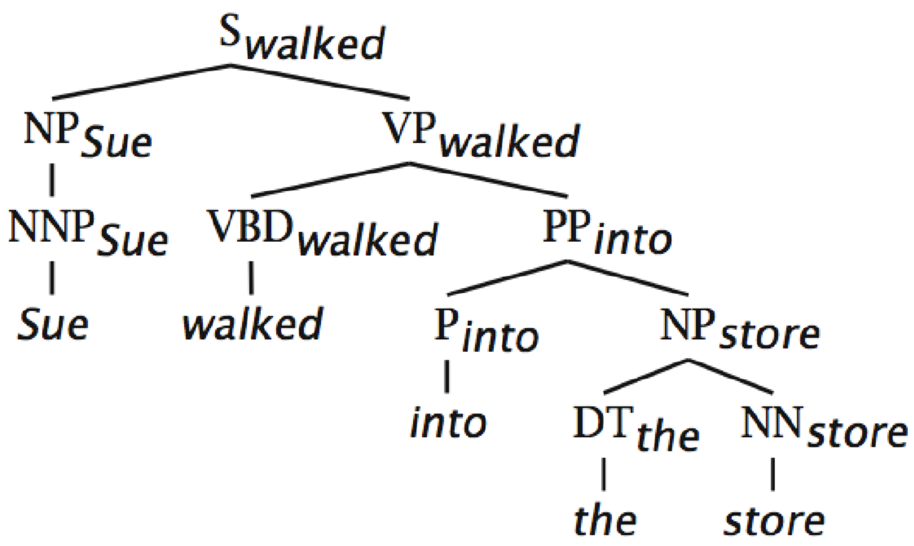
Dependency Structures

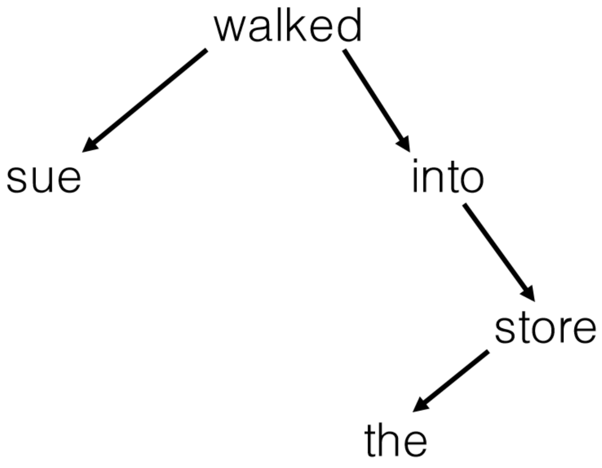
- Instead of annotating what phrases are there, annotate the head words of each constituent
- Syntactic structure consists of:
  - Lexical items
  - Binary asymmetric relations → dependencies
- The edges form a directed tree



Dependency and Constituency Structures

- Dependency grammars have heads
  - Not native to constituency trees
- With head rules → extract dependency structure from constituency tree
  - Other direction is trickier!





Headed Phrase Structure

- Phrases are often headed by particular word types with some modifiers:
  - VP →… Verb …
  - NP → … Noun …
  - ADJP → … Adjective …
  - ADVP → … Adverb …
- **This captures a dependency**

Example Annotation in Dependency and Constituency Format (in English)

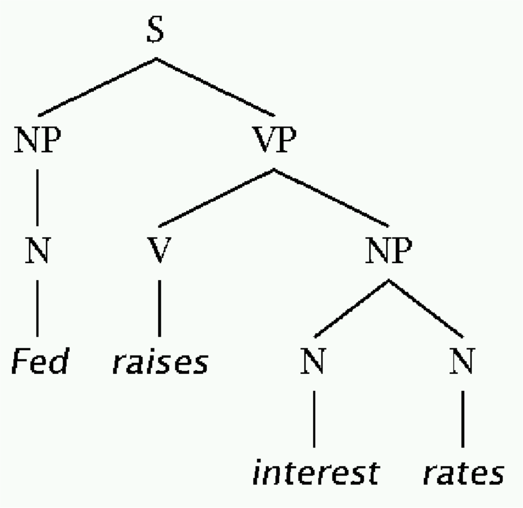
- Transitive clauses:
  - John kicked the ball
  - John took a book yesterday
- Copula clauses:
  - John is tall
- Complement clauses:
  - John says he’s rich
  - John wants to go

Example Annotation in Dependency and Constituency Format (in English)

- Relative Clauses:
  - The dog that ate my homework
- Subordinate clauses:
  - After John came home, he took a shower
- Coordination:
  - John and Mary got married

Constituency Parsing and Context-free Grammars (CFGs)

- Writing parsing rules:
  - N → Fed
  - V → raises
  - NP → N
  - S → NP VP
  - VP → V NP
  - NP → N N
  - NP → NP PP
  - N → interest
  - N → raises



Writing the Rule of Grammar: Context-free Grammars

- A context-free grammar is a tuple $<N, \Sigma, S, R>$
  - N : the set of non-terminals
    - Phrasal categories: S, NP, VP, ADJP, etc.
    - Parts-of-speech (pre-terminals): NN, JJ, DT, VB
  - Σ : the set of terminals (the words)
  - S : the start symbol
    - Often written as ROOT or TOP
    - ***Not*** usually the sentence non-terminal S – why not?
  - R : the set of rules
    - Of the form $X \rightarrow Y_1 Y_2 \ldots Y_n$, with $X \in \boldsymbol{N}$, $\boldsymbol{n \geq 0}$, $\boldsymbol{Y_i} \in (\boldsymbol{N} \cup \Sigma)$
    - Examples: $S \rightarrow NP\ VP$, $VP \rightarrow VP\ CC\ VP$
    - Also called rewrites, productions, or local trees

Example Grammar

N = {S, NP, VP, PP, DT, Vi, Vt, NN, IN}
S = S
Σ = {sleeps, saw, man, woman, telescope, the, with, in}

$R =$

S	$\Rightarrow$	NP	VP
VP	$\Rightarrow$	Vi	
VP	$\Rightarrow$	Vt	NP
VP	$\Rightarrow$	VP	PP
NP	$\Rightarrow$	DT	NN
NP	$\Rightarrow$	NP	PP
PP	$\Rightarrow$	IN	NP

Vi	$\Rightarrow$	sleeps
Vt	$\Rightarrow$	saw
NN	$\Rightarrow$	man
NN	$\Rightarrow$	woman
NN	$\Rightarrow$	telescope
DT	$\Rightarrow$	the
IN	$\Rightarrow$	with
IN	$\Rightarrow$	in

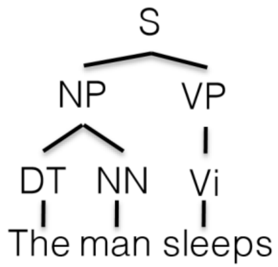
S=sentence, VP-verb phrase, NP=noun phrase, PP=prepositional phrase, DT=determiner, Vi=intransitive verb, Vt=transitive verb, NN=noun, IN=preposition

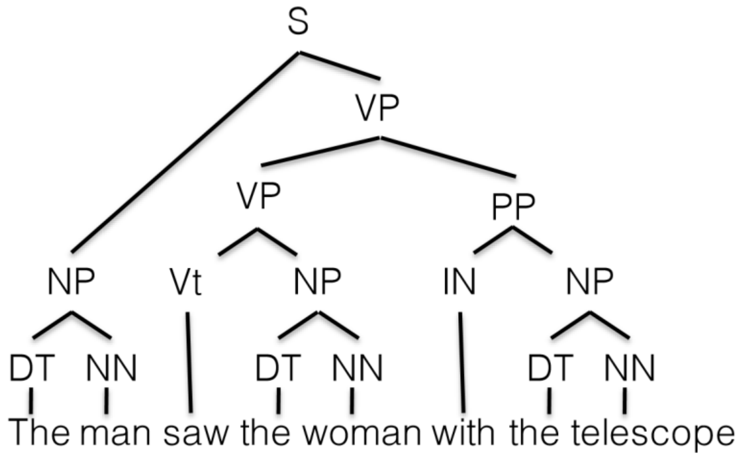
Example Parse

$R =$

S	$\Rightarrow$	NP	VP
VP	$\Rightarrow$	Vi	
VP	$\Rightarrow$	Vt	NP
VP	$\Rightarrow$	VP	PP
NP	$\Rightarrow$	DT	NN
NP	$\Rightarrow$	NP	PP
PP	$\Rightarrow$	IN	NP

Vi	$\Rightarrow$	sleeps
Vt	$\Rightarrow$	saw
NN	$\Rightarrow$	man
NN	$\Rightarrow$	woman
NN	$\Rightarrow$	telescope
DT	$\Rightarrow$	the
IN	$\Rightarrow$	with
IN	$\Rightarrow$	in





S=sentence, VP-verb phrase, NP=noun phrase, PP=prepositional phrase, DT=determiner, Vi=intransitive verb, Vt=transitive verb, NN=noun, IN=preposition

A Parse Tree as a CFG Derivation

- Deterministic Parsing:
  - Given a grammar (say a CFG grammar)
  - Given a sentence
  - Find a derivation that yields the sentence in its leaves

- This can be done with dynamic programming; we will see a more general solution to this problem next lesson