Syntactic Analysis

- Extracting information from text requires syntactic analysis of words, phrases, and clauses.
- The most common syntactic analyzers are:
  - part-of-speech taggers
  - shallow parsers (sometimes called "chunkers")
  - full parsers
- There are many different types of parsers, but the most commonly used are *constituency parsers* and *dependency parsers*.

Morphology & Stemming

- Morphological analyzers decompose words into roots and affixes (prefixes or suffixes in English).

  kick : kicks, kicked, kicking

  virus : viruses, antivirus

  happy : happily, happiness, unhappy

- Stemmers reduce a word down to a stem, for the purpose of generalizing to similar words, but the stem may not be a proper linguistic root.

  pollut : pollute, polluter, polluters, pollution

Part of Speech Tagging

Part-of-speech (POS) tagging systems assign POS tags to the words in a sentence based on their context.

The_{ART} armed_{ADJ} man_{NN} shot_{VB} the_{ART} wild_{ADJ} bear_{NN}.

The part-of-speech for a word depends entirely on its syntactic function in a particular context!

The $\textbf{light}_{\textbf{ADJ}}$ blue candle will $\textbf{light}_{\textbf{VB}}$ the room until the ceiling $\textbf{light}_{\textbf{NOUN}}$ is repaired.

Basic Parts of Speech

- **Parts of Speech Classes:** adjective, adverb, article, conjunction, noun, verb, preposition, pronoun, etc.
- A **closed class** contains a relatively fixed set of words; new words are rarely introduced into the language

  e.g., articles, conjunctions, pronouns, prepositions, …

- An **open class** contains a constantly changing set of words; new words are *often* introduced into the language.

  adjectives, adverbs, nouns, verbs

(Some) Closed Class Parts-of-Speech

Articles: a, an, the

Conjunctions: and, but, or, ...

Demonstratives: this, that, these, ...

Prepositions: to, for, with, between, at, of, ...

Pronouns: I, you, he, she, him, her, myself, ...

Quantifiers: some, every, most, any, both, ...

Open Class Parts-of-Speech

(tag names from the Penn Treebank)

Nouns: represent objects, places, concepts, events. Examples:

NN = common, singular NNS = common, plural
NNP = proper noun, singular NNPS = proper noun, plural

Verbs: represent activities, commands, assertions. Tenses often yield different verb forms and POS tags. Examples:

VBN = past participle, VBD = past tense, VBG = present participle
MD = modal auxiliary verb

Adjectives: attributes that typically modify nouns or act as predicate adjectives (e.g., "I am happy").

Adverbs: can modify verbs, adverbs, adjectives, and clauses

Active vs. Passive Voice

Passive voice consists of a form of `be' followed by a past participle verb form.

Active Voice	**Passive Voice**
I saw him.	*He was seen by me.*
I will find him.	*He will be found by me.*
I have found him.	*He has been found by me.*

Important: the roles are reversed in active and passive voice!

John killed Sam.	Subject is killer. Direct Object is victim.
Sam was killed by John.	Subject is victim. Object of `by' PP is killer.

Transitivity

Transitive verbs require <u>*syntactic*</u> NP objects.

- An *intransitive* verb has no object.

  she laughed, he lied

- A *transitive* verb has a **direct object**.

  she ate an apple, he read a book

- A *ditransitive/bitransitive* verb has two NPs: a **direct object** and an **indirect object**.

  he gave his mom a gift

  she sang the baby a song

Parsing

A parsing algorithm determines the syntactic structure of a sentence with respect to a grammar.

NLP systems typically use *context-free grammars*.

Simple example:

S -> NP VP	VP -> VP1
NP -> art NP1	VP -> VP1 PP
NP1 -> adj NP1	VP1 -> verb
NP1 -> NP2	VP1 -> verb NP
NP2 -> noun	VP1 -> verb NP NP
NP2 -> noun NP2	PP -> prep NP
NP2 -> noun PP	

Shallow Parsing

- *Shallow parsers* (sometimes called *partial parsers* or *chunkers*) identify local syntactic constituents.
- Shallow parsers typically identify base NPs, VPs, PPs, and sometimes ADJPs.
- There is no tree structure for the entire sentence, and usually no links (or limited links) between constituents.
- Shallow parsers are relatively easy to build, work quite well, and are typically fast.

Shallow Parsing Examples

[The quick brown fox]$_{NP}$ [jumped]$_{VP}$ [over the lazy dog]$_{PP}$

[The mayor]$_{NP}$ [of Salt Lake City]$_{PP}$ and [the president]$_{NP}$ [of the teacher's union]$_{PP}$ [teamed up]$_{VP}$ [in budget negotiations]$_{PP}$ [on Tuesday]$_{PP}$

Shallow Parsing with Rules

Shallow parsers can be easily built by defining a simple grammar for basic syntactic constituents, such as NPs, VPs, etc.

A common approach is to encode the grammar in finite-state machines, sometimes cascaded FSMs.

Some parsing algorithms can also be used to do shallow parsing with a grammar, such as bottom-up chart parsing.

Shallow Parsing as Classification

- Shallow parsing can also be viewed as a classification task.
- Machine learning classifiers can be trained to identify different types of syntactic phrases.
- The most common scheme is **IOB labeling**:

  I = inside, O = outside, B = beginning

Example for NP chunking:

Susan$_B$ Miller$_I$ gave$_O$ Rover$_B$ two$_B$ dog$_I$ biscuits$_I$ as$_O$ a$_B$ treat$_I$.

Pros and Cons of Shallow Parsing

\+ Shallow parsing is typically much faster than full parsing.

\+ Shallow parsing can be quite accurate, and sufficient for many application tasks.

\+ Shallow parsers are more robust given ungrammatical and ill-formed input.

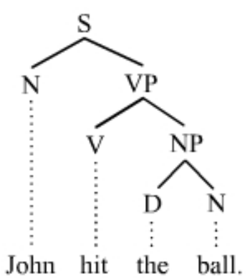
– But shallow parsers provide substantially less syntactic information, e.g. no PP attachments.

– Shallow parsers will often make mistakes for nested (embedded) structures, such as relative clauses.

Constituency-based Parse Trees

A **Parse Tree** represents a sentence's *phrase structure* with respect to a grammar.

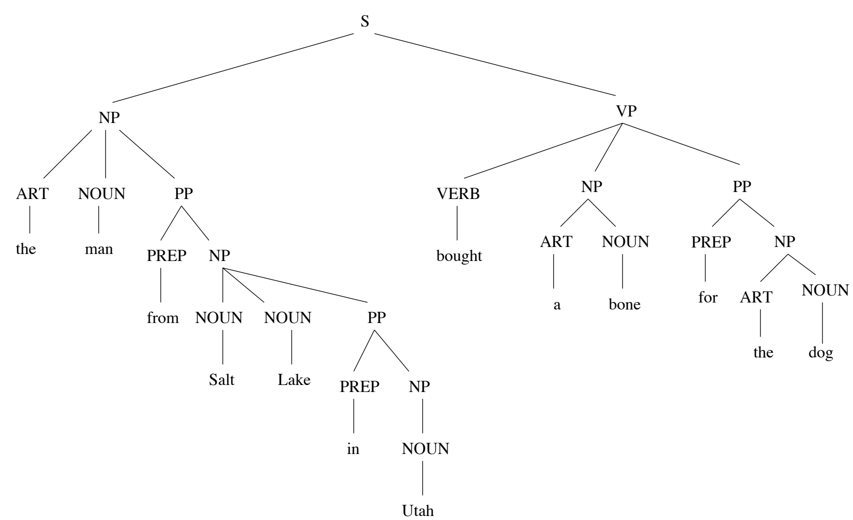
Phrase structure parsers generate *constituency-based* parses.



Constituency-based parse tree

(image from Wikipedia)

Bigger Parse Tree



Multiple Parse Trees

Mary bought Milkbone treats.

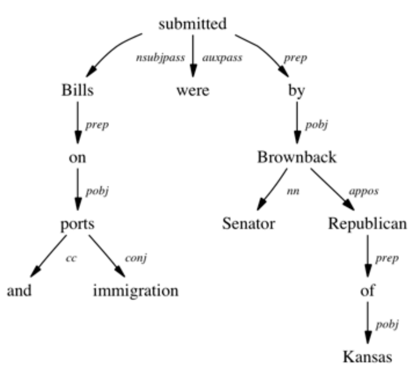
Dependency Parsing

- A dependency parse representation is essentially a directed graph of grammatical relations.
- The parse is often decomposed into pairwise dependencies based on the edges in the graph.
- Relations are between a word (a governing head) and its dependents.
- A dependency parse can be generated directly, or produced as a transformation from a phrase structure parse.

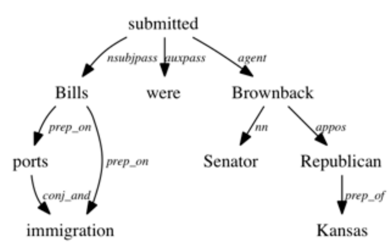
Dependency Parse Graph

Examples of Stanford dependency parser's output:

Figure 2. Basic dependencies



Stanford parser's collapsed dependencies:



Types of Dependency Relations

- Different parsers represent different dependency relations, just like different constituency-based parsers use different phrase structure grammars.
- The Stanford dependency parser represents about 50 grammatical relations.
- Each dependency is a binary relation between a governor (head) and its dependent.

Examples of Dependency Relations

nsubj : nominal subject (syntactic subject of clause)

"Clinton defeated Dole" **nsubj**(defeated, Clinton)

"The baby is cute" **nsubj**(cute, baby)

nsubjpass : passive nominal subject (syntactic subject of passive clause)

"Dole was defeated by Clinton" **nsubjpass**(defeated, Dole)

agent : passive verb complement with preposition "by"

"He was killed by police" **agent**(killed, police)

Examples of Dependency Relations

dobj: direct object of a VP

"She gave Rover a treat" **dobj**(gave, treat)

iobj: indirect object of a VP

"She gave Rover a treat" **iobj**(gave, Rover)

pobj: object of a preposition (head of NP following the preposition)

prep: head of phrase that the preposition attaches to

"Rover ate the cookies on the kitchen table"

pobj(on, table)

prep(on, cookies)

Examples of Dependency Relations

det: determiner and head of its NP

amod: adjectival modifier

advmod: adverbial modifier

"the genetically modified food"

det(food, the)

advmod(modified, genetically)

amod(food, modified)

Examples of Dependency Relations

appos: NP immediately to the right of another NP in an appositive structure

"Steve Ballmer, CEO" **appos**(Ballmer, CEO)

"Steve Ballmer (CEO)" **appos**(Ballmer, CEO)

infmod: infinitive that modifies an NP

"a plan to graduate" **infmod**(plan, graduate)

xcomp: infinitive that modifies a VP or ADJP

"He likes to swim" **xcomp**(likes, swim)

"He is ready to swim" **xcomp**(ready, swim)

Parsing Tools

- Many syntactic analysis tools are freely available, including part-of-speech taggers, chunkers, constituency-based parsers, and dependency parsers.
- Some of the most well-known toolkits are:
  - OpenNLP
  - Stanford NLP group
  - LingPipe
  - NLTK
  - GATE

Beware of Domain Differences

- Most NLP systems today are trained with annotated data using stastical methods and machine learning.
- And they are usually trained on news articles!
- Performance on substantially different types of text can be dramatically different due to:
  - polysemy (e.g., `*share'* can be a verb or noun)
  - unknown vocabulary
  - different frequent structures and idiosyncracies
- The ideal solution is to retrain the tool using domain-specific texts.