Spoken Human Robot Interaction

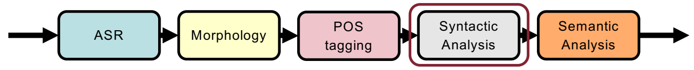
Syntactic Analysis

SAPIENZA
UNIVERSITÀ DI ROMA

Andrea Vanzo, **Daniele Nardi**
lastname@dis.uniroma1.it
Department of Computer, Control, and Management Engineering
"Sapienza" University of Rome, Italy

Artificial Intelligence
AY 2018/19

Our Command Interpretation pipeline



- Moving from words to larger units
  - Syntax and Grammars
- Why should one care?
  - Grammars (and parsing) are key components in many NLP applications

Syntax Applications

- Syntax checking in programming languages
  - Check syntactic correctness
- Most NLP tasks
  - Feature generation for NLP ML-based tasks
- Machine Translation
  - Language to Language syntactic mapping

Syntax aims

- Study the relational structures among words
- Define how words are combined to generate "correct" sentences

Go to the kitchen ✓

The to kitchen go ✗

Syntax Basics

- **Grammar**: declarative model that defines a language
- **Parsing**: process of deciding whether a sentence belongs to the language
  - A syntactic structure can then be associated to the sentence

Outline

- Constituent-based representation
  - Chomsky hierarchy
  - Ambiguities
  - Parsing
    - Deterministic vs Statistical
- Dependency-based representation
  - Dependency relations
  - Parsing

Constituent-based representations

- Based on the concept of **constituent**
- Constituent: span of words (*syntagm*) composing an atomic syntactic structure

[Bring] [me] [the mug] [from the table]

Constituents

- Sentences (S)
  - [the book is on the table]
- Noun Phrases (NP)
  - [the book] is on the table
- Verb Phrases (VP)
  - the book [is] on the table
- Prepositional Phrases (PP)
  - the book is [on the table]
- …

Needed to properly express the structure of sentences:
e.g. NP followed by VP

How to recognize a constituent?

- A constituent can be:
  - Moved: you can move a constituent while preserving a correct syntactic structure
    - Take the mug from the table **[to the tray]** → Take the mug **[to the tray]** from the table
  - Isolated: you can ask questions about a constituent of the sentence
    - Take **[the mug]** from the table to the tray → What do I need to move? [the mug]
  - Coordinated
    - Take **[the mug]** and **[the glass]** from the table to the tray

What is the goal of a syntactic model?

- Identify constituents
  - [Take] [the mug] [to the kitchen]
- Recognize the correct syntactic structure

$[[\text{Take}]_{Verb}\ [[\text{the}]_{Det}\ [\text{mug}]_{Noun}]_{NP}\ [[\text{to}]_{Prep}\ [[\text{the}]_{Det}\ [\text{kitchen}]_{Noun}]_{NP}]_{PP}]_{S}$

The theoretical foundation

Noam Chomsky
- Born December 7, 1928
- Currently Professor Emeritus of linguistics at MIT
- Created the theory of generative grammars

Probably the main contributor to modern linguistics

Preliminary Definitions

- **Language**
  - "A language is a collection of sentences of finite length all constructed from a finite alphabet of symbols."
- **Grammar**
  - "A grammar can be regarded as a device that enumerates the sentences of a language."

- A **grammar** generates a **language**
- A **language** is defined by a **grammar**

- Thus, a **language L** is the set of "strings" that can be generated by the **grammar G**

Grammars

A grammar is a tuple G = (N, Σ, P, S) where:

- N is a finite set of non-terminals
- Σ is a finite set of terminals and is disjoint from N
- P is a finite set of production rules of the form
  - $w \in (N \cup \Sigma)^* \rightarrow w \in (N \cup \Sigma)^*$
- $S \in N$ is the start symbol

Definition of language through a grammar

Let $(A \rightarrow \beta)$ be a production rule and $\alpha, \gamma \in \{N \cup \Sigma\}^*$ then
$\alpha A \gamma$ **directly derives** $\alpha\beta\gamma$

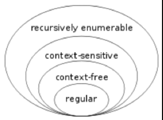
Let $\alpha_1, \ldots, \alpha_n$ be a sequence of direct derivations such that α_i directly derives α_{i+1} then
α_1 **derives** α_n

The **language generated by a grammar** is the set of strings of terminal symbols that can be derived from the start symbol *S*.

The Chomsky hierarchy

- Languages have different complexity, depending on the production rules

Grammar	Languages	Automaton	Production rules
Type-0	Recursively enumerable	Turing machine	$\alpha \rightarrow \beta$, $\lvert\alpha\rvert \geq 1$ $\lvert\beta\rvert \geq 0$
Type-1	Context-sensitive	Linear-bounded non-deterministic Turing machine	$\alpha \rightarrow \beta$, $\lvert\alpha\rvert \geq 1$ $\lvert\beta\rvert \geq 1$ $\lvert\alpha\rvert \leq \lvert\beta\rvert$
Type-2	Context-free	Non-deterministic pushdown automaton	$A \rightarrow \beta$, $\lvert\beta\rvert \geq 1$
Type-3	Regular	Finite state automaton	$A \rightarrow s$ or $A \rightarrow sB$



$s \in \Sigma$
$A, B \in N$
$\alpha, \beta \in \{N \cup \Sigma\}^+$

The Chomsky hierarchy vs NLP

- Context-sensitive languages
  - German
  - Dutch
  - Turkish
- Context-free languages
  - English
  - Italian
  - Most of programming languages
- Regular languages
  - Most of the dialogic interactions

Context-free grammar for NL

- **Constituents** and **POS tags** → N (non-terminal)
  - POS tags are pre-terminals
- **Words** → Σ (terminal)
- **Syntactic rules** → P (productions)
- **Sentence** → S (starting symbol)

S → NP VP
NP → Det Noun
VP → Verb

Det → a | the
Noun → flight | train
Verb → left

Lexicon

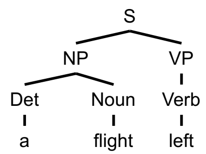
N = {S, NP, VP, Det, Noun, Verb}

Σ = {a, the, flight, train, left}

L_G = {a flight left, a train left, the flight left, the train left}

Syntactic tree

- Syntactic structure is usually represented as a tree



Bracketed notation:

[**S** [**NP** [**Det** a] [**Noun** flight]] [**VP** [**Verb** left]]]

Parsing is the process of mapping a sentence into (one of) its syntactic tree → ambiguity

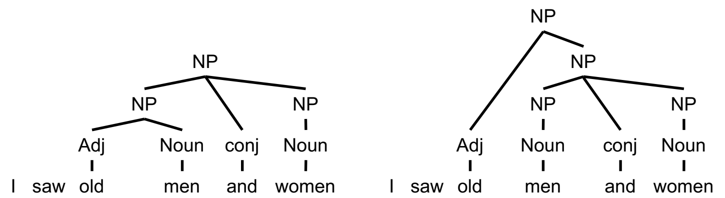
Syntactic ambiguity

One morning I shot an elephant in my pajamas.
How he got into my pajamas I don't know.

Groucho Marx, Animal Crackers, 1930

Syntactic ambiguity

- A sentence that has more than two syntactic trees is ***ambiguous***



Syntactic ambiguity

- Coordination ambiguity
  - different sets of phrases can be conjoined by a conjunction

*I saw old men **and** women*

- Attachment ambiguity
  - a constituent can be attached to the parse tree at more than one place

*We saw the Eiffel Tower **flying to Paris***

Type of syntactic structures

- Declarative
  - S → NP VP *A flight left*
- Imperative
  - S → VP *Stop!*
- Interrogative (Yes/No)
  - S → Aux NP VP *Does the bus stop here?*
- Interrogative (Wh)
  - S → Wh-NP VP *Which flights landed tonight in FCO?*
  - S → Wh NP Aux NP VP *Which flight do you prefer?*

The Noun Phrase: NP → Det Nominal

- The determiner: *a glass, …, Daniele's glass,*
  - Det → simpledet | NP's VP | "none"
- The Nominal:
  - Nominal → Before-Noun Noun After-Noun
- Before the Noun: *two friends, the tallest girl, …*
  - Cardinal/ordinal numbers, quantifiers, adjectives
- After the Noun:
  - Prepositional phrases *all people from Spain*
  - Non-finite clauses *students arriving from SPV*
  - Relative clauses *colleagues that do research*

The Verb Phrase: NP → Verb [NP] [PP]

Many variations beyond the simple structure:

Sentential complements: *I want to go home*
- VP → Verb S

Compatibility verb-complement:

- Classical Transitive/Intransitive
- Modern English 100+ Subcategories
  - Verb-with-NP-complement → find | leave | …
  - Verb-with-S-complement → think | believe | …

Coordination

Conjunctions like *and, or,but*:

- NP → NP *and* NP
  - *I have the bag and the PC*
- Nominal → Nominal *and* Nominal
  - *I know the title and editor*
- VP → VP *and* VP
  - *The flights leaving FCO and arriving in JFK*
- S → S *and* S
  - *I like pears and I buy them*

X → X *and* X

Dealing with morphology

The book **is** on the table
The book **are** on the table

- Two solutions:
  - Grammar expansion

S → 3sgNP 3sgVP
3sgNP → Det sgNoun

  - Parametrization

S(x) → NP(x) VP(x)
NP(x) → Det Nom(x)
Nom(x) → Nom(x) Noun(x) | Noun(x)

Noun(sg) → flight | plane ...
Noun(pl) → flights | planes ...
V(sg) → leaves | ...
V(pl) → leave | ...

Treebanks

Treebanks are linguistic resources that provide corpora of syntactically annotated sentences (needed for ML parsing)

```
((S
   (NP-SBJ (DT That)
     (JJ cold) (, ,)
     (JJ empty) (NN sky) )
   (VP (VBD was)
     (ADJP-PRD (JJ full)
       (PP (IN of)
         (NP (NN fire)
           (CC and)
           (NN light) ))))
   (. .) ))
```
(a)

```
((S
   (NP-SBJ The/DT flight/NN )
   (VP should/MD
     (VP arrive/VB
       (PP-TMP at/IN
         (NP eleven/CD a.m/RB ))
       (NP-TMP tomorrow/NN )))))
```
(b)

Example from PennTreebank

Parsing for Constituency-based representation

Parsing*(string)* is a decision procedure that returns:
- **yes** (and the corresponding parsing tree), if *string* belongs to the language,
- **no,** otherwise.

Different possible solutions:

- Deterministic
  - Grammar
- Statistical
  - Grammar + probabilities

Deterministic parsing

It relies just on the predefined grammar and on an algorithm

```
S → NP VP
S → Aux NP VP
S → VP
NP → Det Noun
NP → PropN
VP → Verb
VP → Verb NP

Det → that | this | a
Noun → book | flight | meal | money
Verb → book |include | prefer
Aux → does
PropN → Houston | TWA
```

Algorithm strategy
- Top-Down VS Bottom-up
- Depth-First VS Breadth-First
- Left-To-Right VS Right-To-Left
- Agenda
- Dynamic Programming

"Does this flight include a meal?"

Deterministic NLP parsers for CFG

- CYK parser (1967)
  - Cocke-Younger-Kasami
  - Requires Normal Form ($A \rightarrow B\ C$ or $A \rightarrow a$)
  - bottom-up and dynamic programming
- Earley parser (from 1968)
  - mixed and dynamic programming

Statistical Constituency-based representation

- Syntactic ambiguity can be alleviated through probabilistic parsing
- Idea: compute the probability of each interpretation and choose the most likely one
- How? Probabilistic Context-Free Grammar (PCFG)

Probabilistic Context-Free Grammars

Probabilistic augmentation of context-free grammars

- Each production rule is paired with a probability

$$P(A \rightarrow \beta)$$

so that

$$\sum_{\beta} P(A \rightarrow \beta) = 1$$

Example of PCFG

Grammar		Lexicon
$S \rightarrow NP\ VP$	[.80]	$Det \rightarrow that$ [.10] | a [.30] | the [.60]
$S \rightarrow Aux\ NP\ VP$	[.15]	$Noun \rightarrow book$ [.10] | $flight$ [.30]
$S \rightarrow VP$	[.05]	| $meal$ [.015] | $money$ [.05]
$NP \rightarrow Pronoun$	[.35]	| $flight$ [.40] | $dinner$ [.10]
$NP \rightarrow Proper\text{-}Noun$	[.30]	$Verb \rightarrow book$ [.30] | $include$ [.30]
$NP \rightarrow Det\ Nominal$	[.20]	| $prefer$ [.40]
$NP \rightarrow Nominal$	[.15]	$Pronoun \rightarrow I$ [.40] | she [.05]
$Nominal \rightarrow Noun$	[.75]	| me [.15] | you [.40]
$Nominal \rightarrow Nominal\ Noun$	[.20]	$Proper\text{-}Noun \rightarrow Houston$ [.60]
$Nominal \rightarrow Nominal\ PP$	[.05]	| NWA [.40]
$VP \rightarrow Verb$	[.35]	$Aux \rightarrow does$ [.60] | can [40]
$VP \rightarrow Verb\ NP$	[.20]	$Preposition \rightarrow from$ [.30] | to [.30]
$VP \rightarrow Verb\ NP\ PP$	[.10]	| on [.20] | $near$ [.15]
$VP \rightarrow Verb\ PP$	[.15]	| $through$ [.05]
$VP \rightarrow Verb\ NP\ NP$	[.05]	
$VP \rightarrow VP\ PP$	[.15]	
$PP \rightarrow Preposition\ NP$	[1.0]	

PCFG for disambiguation

- Given a parse tree T and a sentence S, their joint probability is defined as

$$P(T,S) = \prod_{i=1}^{n} \mathrm{P}(RHS_i|LHS_i)$$

- Generate all the parse trees and choose the one with highest probability

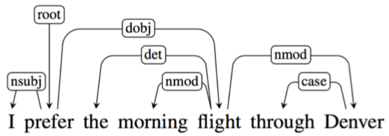
$$T^*(S) = \operatorname*{argmax}_{T:T\rightarrow S} P(T,S)$$

CYC parsers can be extended

Dependency-based representation

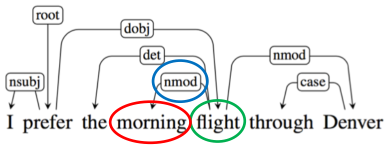
- Used in most of the modern syntactic parsers
  - Particularly suited for languages with too much flexibility of words order
    - Czech
  - Closer to semantic formalisms
- Focus:
  - Constituent-based grammars: constituents (or syntagms)
  - **Dependency-based grammars: dependency relations among words of the sentence**

Dependency-based representation



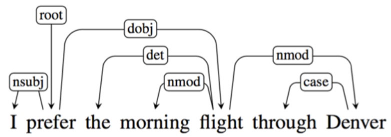
- Tree nodes are words (instead of constituents)
- Each connecting (directed!!) arc is labeled with the type of syntactic relationship
  - drawn from a fixed dictionary of grammatical relations

Dependency relations



- Each dependency relation is composed of:
  – Label: defines the nature of the dependency relation
  – Head: the word governing the relation (main role of a constituent)
  – Dependent: the direct/indirect dependent of the relation (rest of the words of a constituent)

Dependency relations



Relation	Description	Relation	Description
root	Word governing the sentence	*case*	Prepositions, postpositions
nsubj	Nominal subject	*conj*	Conjunct
dobj	Direct object	*iobj*	Indirect object
nmod	Nominal modifier	*amod*	Adjectival modifier
det	Determiner	…	…

Parsing dependency-based representation

There are two main approaches to dependency parsing:

– Dynamic Programming:
Optimization-based approaches that search a space of trees for the tree that best matches some criteria
  - Treat dependencies as constituents, algorithm similar to CKY
  - Examples: MST (Ryan McDonald), Bhonet's parser

– Deterministic parsing:
Shift-reduce approaches that take actions based on the current word and state (use classifier to predict next parsing step)
  - Example: Malt parser (Joakim Nivre)

– More recently replaced by NN (e.g. SPACY)

Tools

- Stanford Parser (provides constituent and dependency trees)
- Charniak Parser (constituent parser with discriminative re-ranker)
- Berkeley Parser (constituent parser with latent variables)
- MST parser (dependency parser, needs POS tagged input)
- Bohnet's parser (dependency parser, needs POS tagged input)
- Malt parser (dependency parser, needs POS tagged input)

References

- Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. https://web.stanford.edu/~jurafsky/slp3/

Chapter 10 (excluding 10.5 and 10.6)

Chapter 11.1

Chapter 12.1

Chapter 13.1,2,3