CSCE 771: Computer Processing of Natural Language

Lecture 6: Projects (Topic Review), Shallow Parsing, Dependency Parsing

PROF. BIPLAV SRIVASTAVA, AI INSTITUTE

6TH SEPTEMBER, 2022

Carolinian Creed: "I will practice personal and academic integrity."

Acknowledgement: Used materials by Profs. Mausam, Jurafsky & Martin, Stanford NLP

Organization of Lecture 6

- Opening Segment
  - Review of Last Lecture
  - Announcements
- Main Lecture
- Concluding Segment
  - About Next Lecture – Lecture 7

Main Section

- Project: complete reviewing of topics
- Shallow Parsing
- Dependency Parsing

Recap of Lecture 5

- We discussed the paper - ”Contextual Word Representations: Putting Words into Computers”, by Noah Smith, CACM June 2020”
- We looked at parsing
  - Roles it plays: verifying , generating, recognizing
  - Many types of parsing: shallow parsing for quick NLP tasks, phrase structure parsing, dependency parsing
- Started reviewing projects – topics

Announcements

- Quiz 1 - next class, in-person and using paper and pen
  - No makeup (best of 3 from 4 quizzes)
  - Will cover concepts discussed in class

Main Lecture

Complete Review of Project Titles

Choosing a Project – Some Considerations

- Scope: what is the problem?
- Current-state: what happens in the problem today?
- Who cares: who will benefit with the problem being solved?
- Desired-state: what will be the future situation if your project succeeds?
- Resources/ dataset: do you have reasonable data and compute resources to do the work?
- Evaluation: how will we measure goodness of the work?

Review project spreadsheet

Discussion: Course Project

- Expectations
  - Apply methods learned in class or of interest to a problem of interest
  - Be goal oriented: aim to finish, be proactive, be innovative
  - Do top-class work: code, writeup, presentation
- Typical pitfalls
  - Not detailing out the project, assuming data
  - Not spending enough time
- What will be awarded
  - Results and efforts (balance)
  - Challenge level of problem

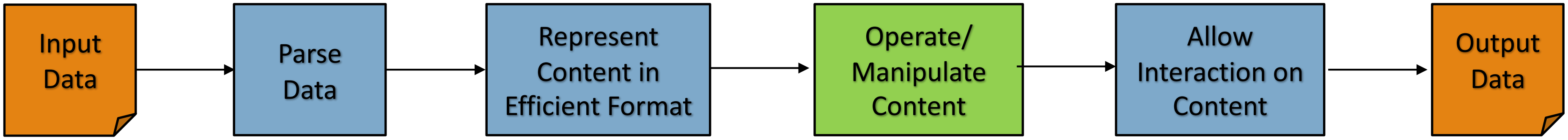
Course Project – Deadlines and Penalty Rubric

What is next ?

- Create project plan and put in your G-drive; project sub-dir; File name: “Project plan”. Extension: .docx or .pdf
- File will contain
  - * Project Title
  - * Description: motivation and expected output
  - * Illustrative Test cases: i.e., Example input / output
  - * Data sources
  - * Technique and tools to use
  - * Metric for measuring output
  - * How will you collect results
  - * Format of report, presentation
  - * Time schedule, by Week
- Penalty: **not** ready by Sep 15, 2022 **[-20%]**

- Other penalties
  - Project report **not** ready by Nov 10, 2022 **[-20%]**
  - Project presentations **not** ready by Nov 15, 2022 **[-10%]**

Review: Parsing


Input Data
Parse Data
Represent Content in Efficient Format
Operate/ Manipulate Content
Allow Interaction on Content
Output Data

Parsing

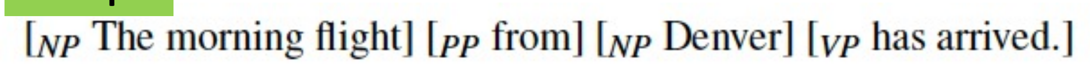
Types of Parsing

- **Phrase structure** / **Constituency Parsing**: find phrases and their recursive structure. Constituency - groups of words behaving as single units, or constituents. Context free grammars are also called Phrase-Structure Grammars
  - **Shallow Parsing/ Chunking**: identify the flat, non-overlapping segments of a sentence: noun phrases, verb phrases, adjective phrases, and prepositional phrases.
- **Dependency Parsing**: find relations in sentences
- **Probabilistic Parsing**: given a sentence X, predict the most probable parse tree Y

Chunking

- Chunking - process of identifying and classifying the flat, non-overlapping segments of a sentence that constitute the basic **non-recursive phrases** corresponding to the major content-word parts-of-speech:
  - noun phrases
  - verb phrases
  - adjective phrases, and
  - prepositional phrases

Example



[$_{NP}$ The morning flight] [$_{PP}$ from] [$_{NP}$ Denver] [$_{VP}$ has arrived.]

- Two operations in this type of parsing:
  - segmenting - finding the non-overlapping extents of the chunks and
  - labeling - assigning the correct tag to the discovered chunks
- Some words may not be part of any chunk

From Jurafsky & Martin

Shallow Parsing/ Chunking

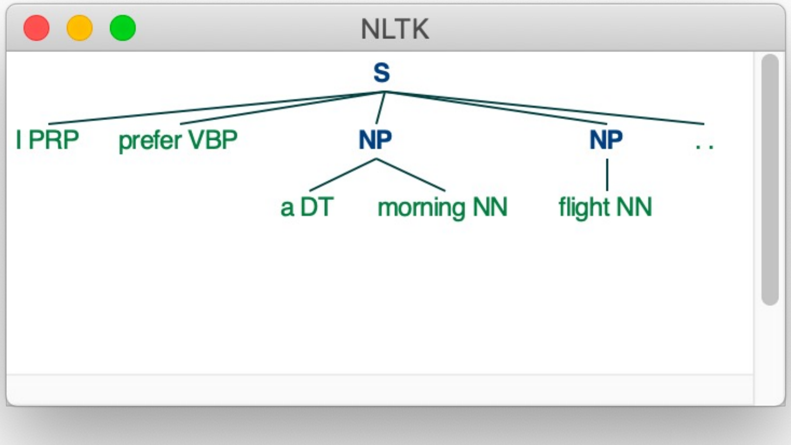
```
data = "I prefer a morning flight."

# Prepare data
tokens = nltk.word_tokenize(data)
tag = nltk.pos_tag(tokens)

# Grammar to use
grammar = "NP: {<DT>?<JJ>*<NN>}"
cp  =nltk.RegexpParser(grammar)

# Parse based on regex
result = cp.parse(tag)
print(result)
```

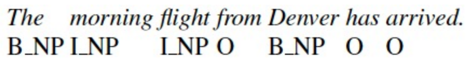
```
(S I/PRP prefer/VBP (NP a/DT morning/NN) (NP flight/NN) ./.)
```



IOB notation

- Chunking - IOB tagging
  - B - beginning of each chunk type
  - I - inside of each chunk type
  - O - one for tokens outside (O) any chunk
- Total: (2N + 1) tags for N chunk types

Example



The morning flight from Denver has arrived.
B_NP I_NP I_NP O B_NP O O

From Jurafsky & Martin

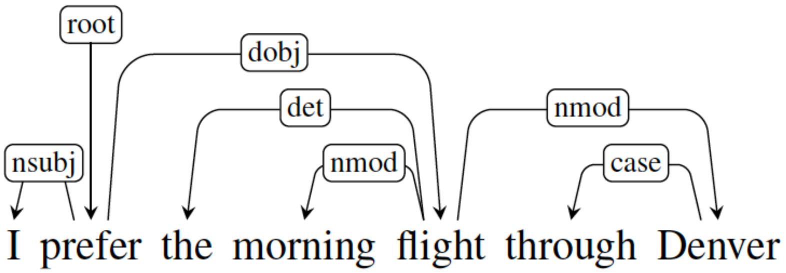
Code and Examples

- Sample code – https://github.com/biplav-s/course-nl-f22/blob/main/sample-code/l6-l7-parsing/Chunking%20-%20syntax%20exploration.ipynb

- Advanced examples – https://www.nltk.org/book/ch07.html

Dependency Parsing

- **Meaning** depends on
  - Words (lemmas) in a sentence
  - Their directed binary grammatical relations with other words
  - (and not on CFGs)
- **Notation**: Labeled arcs are from heads to dependents



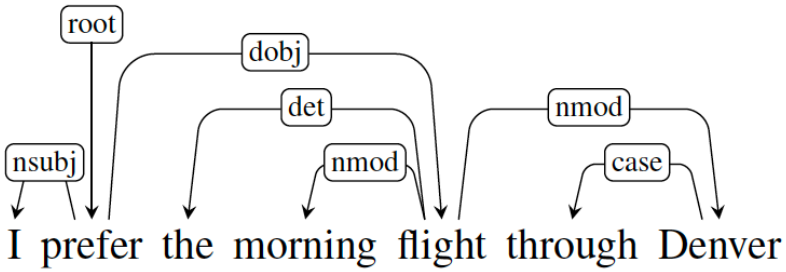
No node corresponding to phrasal constituents or lexical categories in the dependency parse

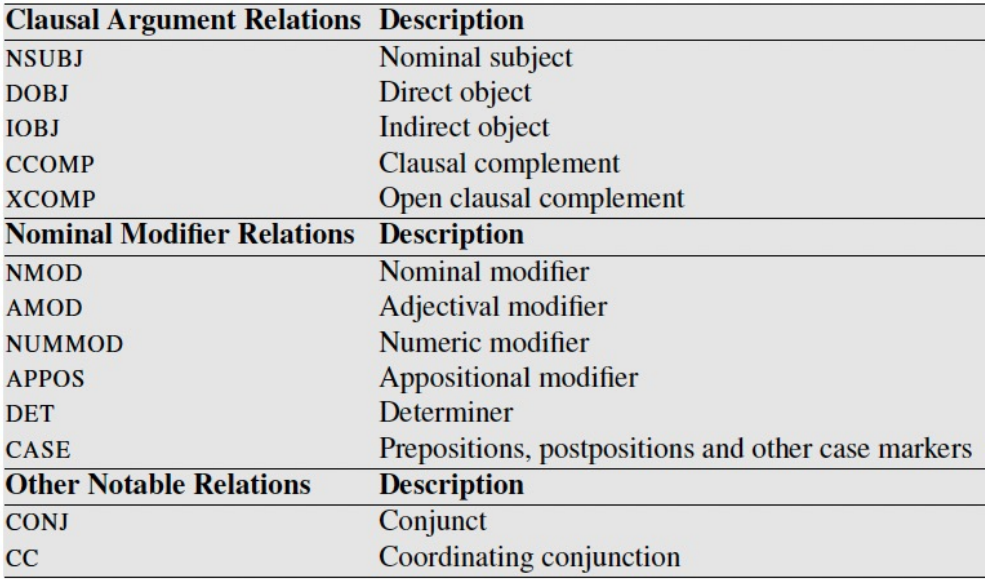
Dependency Conditions

1. There is a single designated root node that has no incoming arcs.
2. With the exception of the root node, each vertex has exactly one incoming arc.
3. There is a unique path from the root node to each vertex in V.

From Jurafsky & Martin

Dependency Parsing



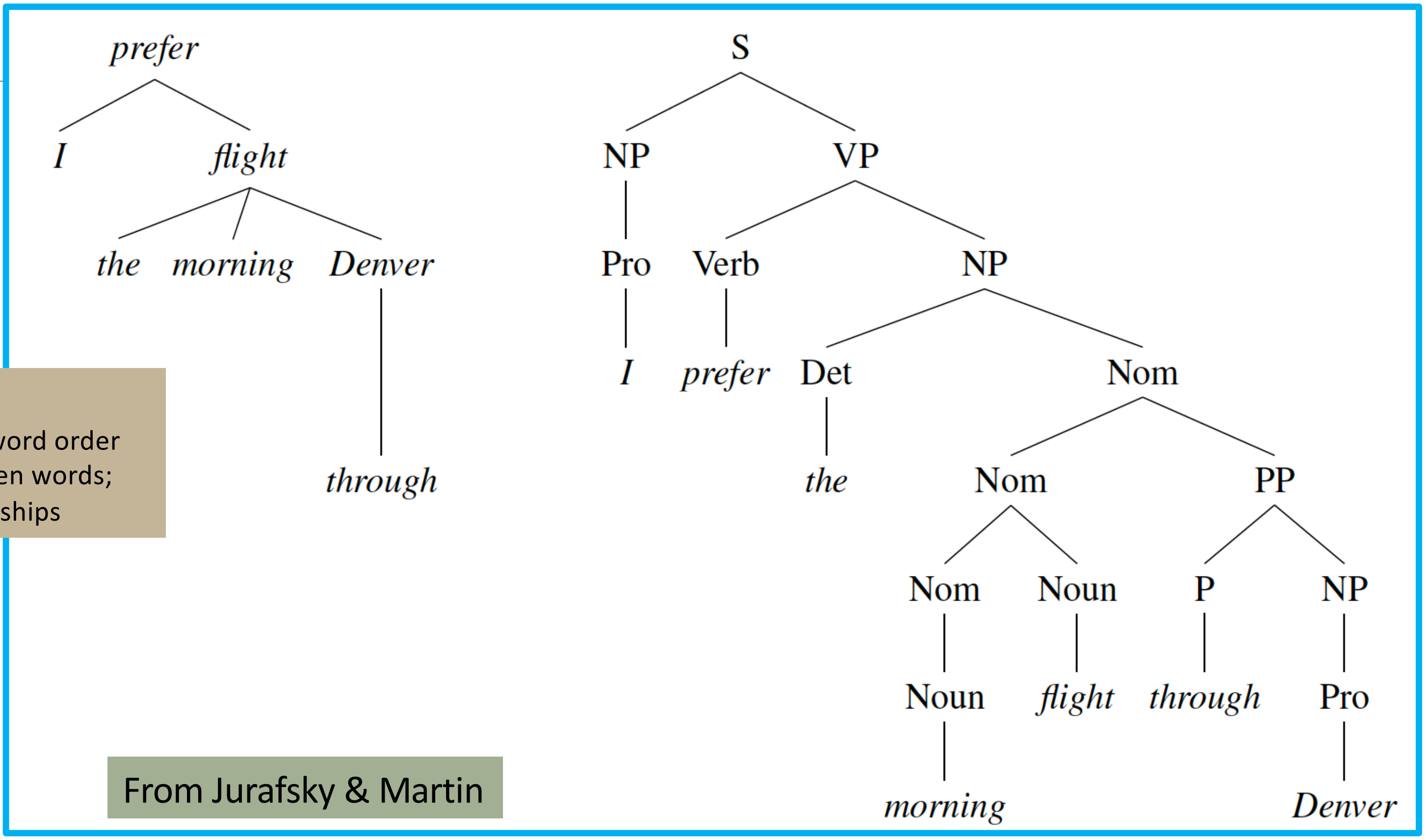


Clausal Argument Relations	**Description**
NSUBJ	Nominal subject
DOBJ	Direct object
IOBJ	Indirect object
CCOMP	Clausal complement
XCOMP	Open clausal complement
Nominal Modifier Relations	**Description**
NMOD	Nominal modifier
AMOD	Adjectival modifier
NUMMOD	Numeric modifier
APPOS	Appositional modifier
DET	Determiner
CASE	Prepositions, postpositions and other case markers
Other Notable Relations	**Description**
CONJ	Conjunct
CC	Coordinating conjunction

Edge: role that the dependent plays with respect to its head. Examples: subject, direct object and indirect object.

From Jurafsky & Martin

Comparison: Dependency and Phrase Structure

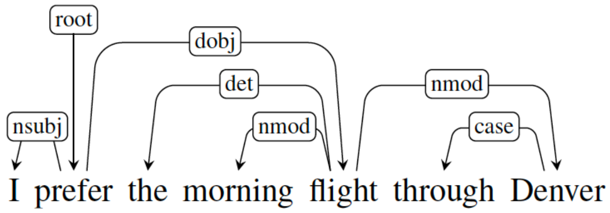


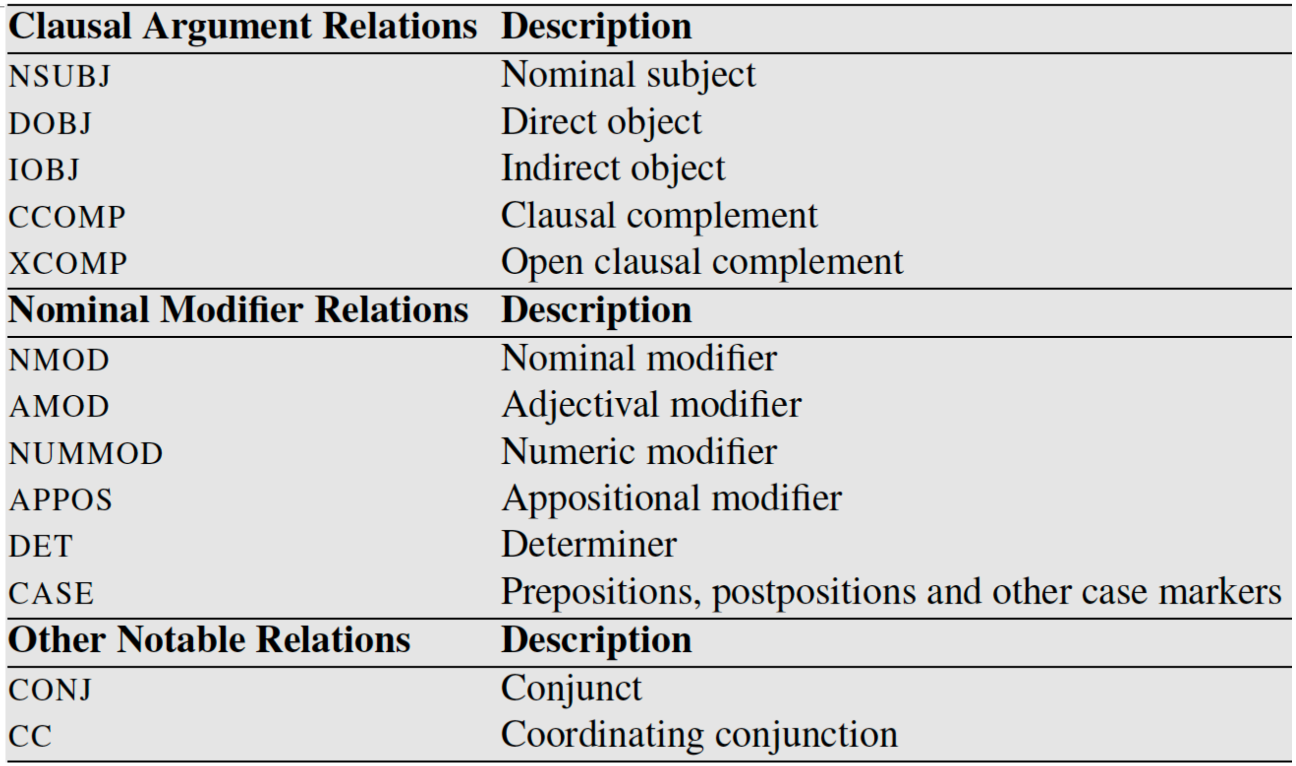
From Jurafsky & Martin

Dependency:

- Useful in languages with free word order
- Highlights relationships between words; useful is tasks needing relationships

Example Dependency Relationships





Clausal Argument Relations	**Description**
NSUBJ	Nominal subject
DOBJ	Direct object
IOBJ	Indirect object
CCOMP	Clausal complement
XCOMP	Open clausal complement
Nominal Modifier Relations	**Description**
NMOD	Nominal modifier
AMOD	Adjectival modifier
NUMMOD	Numeric modifier
APPOS	Appositional modifier
DET	Determiner
CASE	Prepositions, postpositions and other case markers
Other Notable Relations	**Description**
CONJ	Conjunct
CC	Coordinating conjunction

From Jurafsky & Martin

Examples of Parsing with Spacy

- Sample code – https://github.com/biplav-s/course-nl-f22/blob/main/sample-code/l6-l7-parsing/parsing%20spacy.ipynb

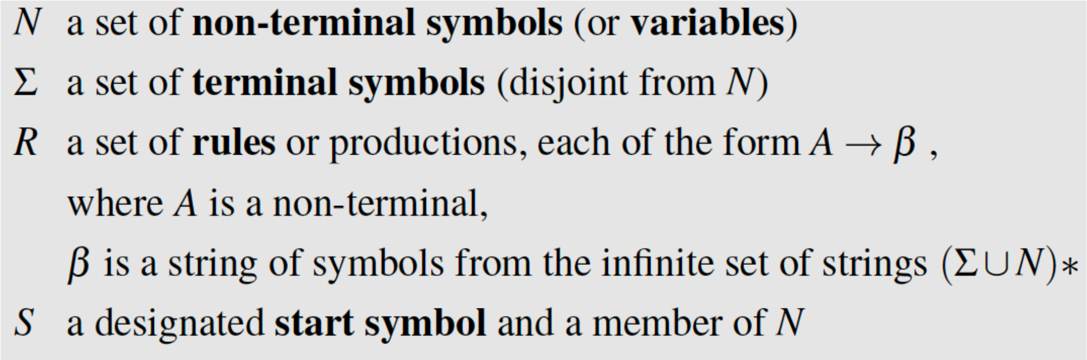
See GitHub

About Grammar Forms

- **Strong equivalent grammars**: Two grammars are strongly equivalent if they generate the same set of strings and if they assign the same phrase structure to each sentence
- **Weakly equivalent grammars**: Two grammars are weakly equivalent if they generate the same set of strings but do not assign the same phrase structure to each sentence.
- **Chomsky Normal Form**: a grammar which is
  - ϵ-free and
  - each production is either of the form A -> B C or A -> a.
- ***Any context-free grammar can be converted into a weakly equivalent Chomsky normal form grammar***
  - A -> B C D becomes
  - A -> B X and X -> C D

From Jurafsky & Martin

Review: Parsing - CFG



N a set of **non-terminal symbols** (or **variables**)

Σ a set of **terminal symbols** (disjoint from N)

R a set of **rules** or productions, each of the form $A \rightarrow \beta$,
where A is a non-terminal,
β is a string of symbols from the infinite set of strings $(\Sigma \cup N)*$

S a designated **start symbol** and a member of N

Example CFG:

- N = {S, NP, VP, }
- Σ = {he, she, walks, sleeps}
- R = {
  - S -> NP, VP
  - NP -> he
  - NP -> she
  - VP -> walks
  - VP -> sleeps

  }
- S = S

Questions: *which strings are in the language of example CFG*
(a) she sleeps (b) walks sheeps (c) sleeps he (d) she walks (e) he and she walks

Parsing Perspective

- **Question**: Is parsing of a sentence unique ?

Example 1: "*Book the dinner flight*"

- Book the flight which has dinner
- Book the flight for dinner

Example 2: *"I made her duck"*

- I cooked duck (**sense**: **animal**) for her
- I cooked duck (**sense: animal**) belonging to her.
- I turned her into duck (**sense**: **animal**)
- I created duck (**sense: object**) for her
- I made her to lower her head or body (**sense: posture**).

From Jurafsky & Martin

Parsing Perspective

- **Question**: Is parsing of a sentence unique ?
- **Answer**: Not necessarily

Issue: Then, which one to return?

Solution: Given a sentence X, predict its parse tree Y

Lecture 6: Concluding Comments

- We reviewed projects
- We reviewed parsers
  - Shallow parsers
  - Dependency parsers

Concluding Segment

About Next Lecture – Lecture 7

Lecture 7

- Statistical parsing
- QUIZ