# Introduction, administrivia
## Parsing
## ISCL-BA-06

Çağrı Çöltekin
/tʃaːrˈɯ tʃœltecˈɪn/
ccoltekin@sfs.uni-tuebingen.de

University of Tübingen
Seminar für Sprachwissenschaft

Winter Semester 2020/21

---

## What is parsing?

- *Parsing* is the task of analyzing a string of symbols to discover its (inherent) structure
- Typically, the structure (and the valid strings in the language) is defined by a *grammar*
- The output of a parser is a structured representation of the input string, often a tree
- *Recognition* is an intimately related task which determines whether a given string is in a language

---

## Ingredients of a parser
### (for natural language parsing)

- A formal grammar defining a language of interest
- An algorithm that (efficiently) verifies whether a given string is in the language (recognizer) and enumerate the grammar rules used for verification (parser)
- A system for ambiguity resolution (very limited coverage in this course)

---

## Grammars

- A grammar is a finite specification of a possibly infinite language
- The most commonly studied type of grammars are *phrase structure grammars*
- Analysis using a (type of) phrase structure grammars result in *constituency* or *phrase structure* trees
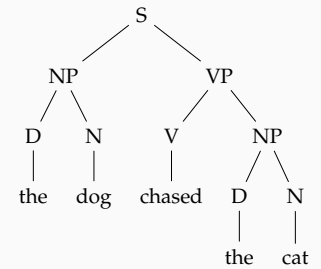


| | | |
|---|---|---|
| S → NP VP | NP → D N | VP → V NP |
| V → chased | D → the | N → cat            N → dog |

---

## Formal languages and natural languages

> *There is in my opinion no important theoretical difference between natural languages and the artificial languages of logicians.*
> — Richard Montague, in "Universal Grammar" (1970)

- Formal grammars are equally important for linguistics as they are important for computer science
- Historically, there has been very strong connections between linguistics and computer science
- The formal languages (that originate in linguistics) has important theoretical consequences for computer science as well

---

## Why study parsing?

- In general, it is a intermediate step for interpreting sentences
- Applications include:
  - Compiler construction
  - Grammar checking
  - Sentiment analysis
  - Information (e.g., relation) extraction
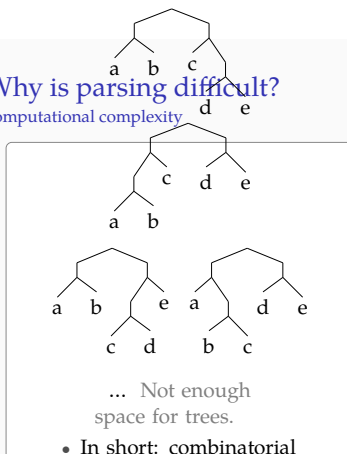  - Argument mining
  - ...

---
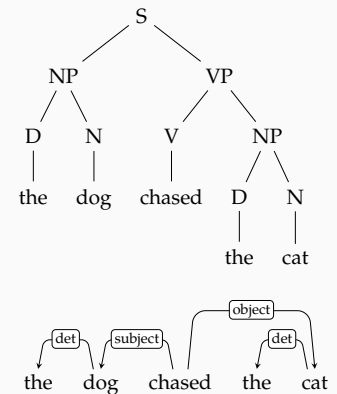
## Why is parsing difficult?
### computational complexity



| words | search space |
|---|---|
| 1 | 1 |
| 2 | 1 |
| 3 | 2 |
| 4 | 5 |
| 5 | 14 |
| 10 | 4862 |
| 15 | 2 674 440 |
| 20 | 1 767 263 190 |
| 25 | 1 289 904 147 324 |

... Not enough space for trees.

- In short: combinatorial expansion.
- Most of what we study in this course is ways to limit this search space based on ...

Fun exercise: try to systematically produce all binary bracketings of a given number of words.

---

## Why is parsing difficult?
### ambiguity (for natural languages) – examples from newspaper headlines

FARMER BILL DIES IN HOUSE

TEACHER STRIKES IDLE KIDS

SQUAD HELPS DOG BITE VICTIM

BAN ON NUDE DANCING ON GOVERNOR'S DESK

PROSTITUTES APPEAL TO POPE

KIDS MAKE NUTRITIOUS SNACKS

DRUNK GETS NINE MONTHS IN VIOLIN CASE

MINERS REFUSE TO WORK AFTER DEATH

Most of the above are lexical ambiguities, but structural ambiguity is also common in natural languages.

## Why is parsing difficult?
more on ambiguities

---

## What is in this course?
A bird's eye view

- Grammars, languages, automata, computation
- Parsing as search: bottom-up, top-down
- Chart parsing: CKY, Earley
- Table driven/deterministic parsing: LL/LR/SLR/GLR parsers
- Probabilistic (context-free) parsing
- Dependency grammars
- Dependency parsing: MST, transition-based parsing

---

## Literature

- **grune2008**. grune2008 (**grune2008**)
- **jurafsky2009**. jurafsky2009 (**jurafsky2009**)
- **kubler2009**. kubler2009 (**kubler2009**)

---

## Practical information

- Lectures Mon/Thu 8:30, online, synchronous via Zoom
- Course web page at https://iscl-parsing2020.github.io/
- The class sessions include lectures and exercises, but exact division is unclear
- Most assignments are mostly pencil-and-paper exercises, there will also be practical assignments, but no programming exercises in this course
- Please obtain a GitHub account if you do not have one. We will use GitHub for some of the exercises (more on this later)
- Please register to the Moodle page of the course, and pay attention to the announcements posted there

---

## Prerequisites

You should have already taken
- Linguistic fundamentals
- Data Structures and Algorithms for CL I
- Data Structures and Algorithms for CL II

  effectively, you need to know some linguistics and formal thinking

  programming skills/knowledge is useful, it is not required for this course

---

## Evaluation

- Final exam at the end of the semester
- Assignments (not graded, but required)
  - (Almost) weekly pencil-and-paper exercises
  - Three bigger, group assignments:
    - Writing a grammar for a subset of English
    - Writing a small constituency treebank
    - Creating a small dependency treebank

---

## Your first assignment

- Your first assignment is available at https://iscl-parsing2020.github.io/a0/
- Please complete as soon as you can (it is easy)
- In summary: introduce yourself, and provide 5 grammatical and 5 ungrammatical sentences
- We will use the data gathered for future practical assignments

---

## Acknowledgments, references, additional reading material

- This set of slides are based on earlier slides by Kurt Eberle, which in turn was based on slides by Helmut Schmid
- Some of the (later) examples are inspired by, or sometimes verbatim borrowings from, the material listed below