UPPSALA
UNIVERSITET

Introduction

Syntactic parsing (5LN713/5LN717)

2020-03-31

Sara Stymne
Department of Linguistics and Philology

Partly based on slides from Marco Kuhlmann

Today

- Introduction to syntactic analysis
- Course information
- Exercises

What is syntax?

- Syntax addresses the question how sentences are constructed in particular languages.
- The English (and Swedish) word *syntax* comes from the Ancient Greek word *sýntaxis* ‘arrangement’.

What is syntax not?

Syntax does not answer questions about …

… how speech is articulated and perceived
(phonetics, phonology)

… how words are formed
(morphology)

… how utterances are interpreted in context
(semantics, pragmatics)

What is syntax not?

Syntax does not answer questions about …

… how speech is articulated and perceived
(phonetics, phonology)

… how words are formed
(morphology)

… how utterances are interpreted in context
(semantics, pragmatics)

simplified

Why should you care about syntax?

- Syntax describes the distinction between well-formed and ill-formed sentences.
- Syntactic structure can serve as the basis for semantic interpretation and can be used for
  - Machine translation
  - Information extraction and retrieval
  - Question answering
  - ...

UPPSALA
UNIVERSITET

Why should you care about syntax?

- Syntactic structure can be useful for analysing large text materials
  - Research in digital humanities
  - Economic analysis

Parsing

The automatic analysis of a sentence with respect to its syntactic structure.

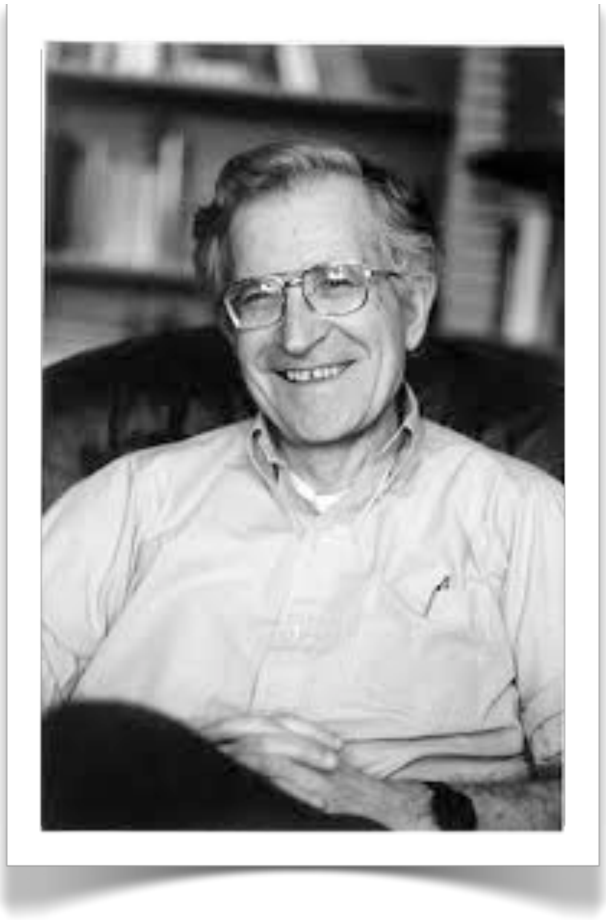
Theoretical frameworks

- Generative syntax
  Noam Chomsky (1928–)
- Categorial syntax
  Kazimierz Ajdukiewicz (1890–1963)
- Dependency syntax
  Lucien Tesnière (1893–1954)

Theoretical frameworks

- Generative syntax
  Noam Chomsky (1928–)
- Categorial syntax
  Kazimierz Ajdukiewicz (1890–1963)
- Dependency syntax
  Lucien Tesnière (1893–1954)

Theoretical frameworks

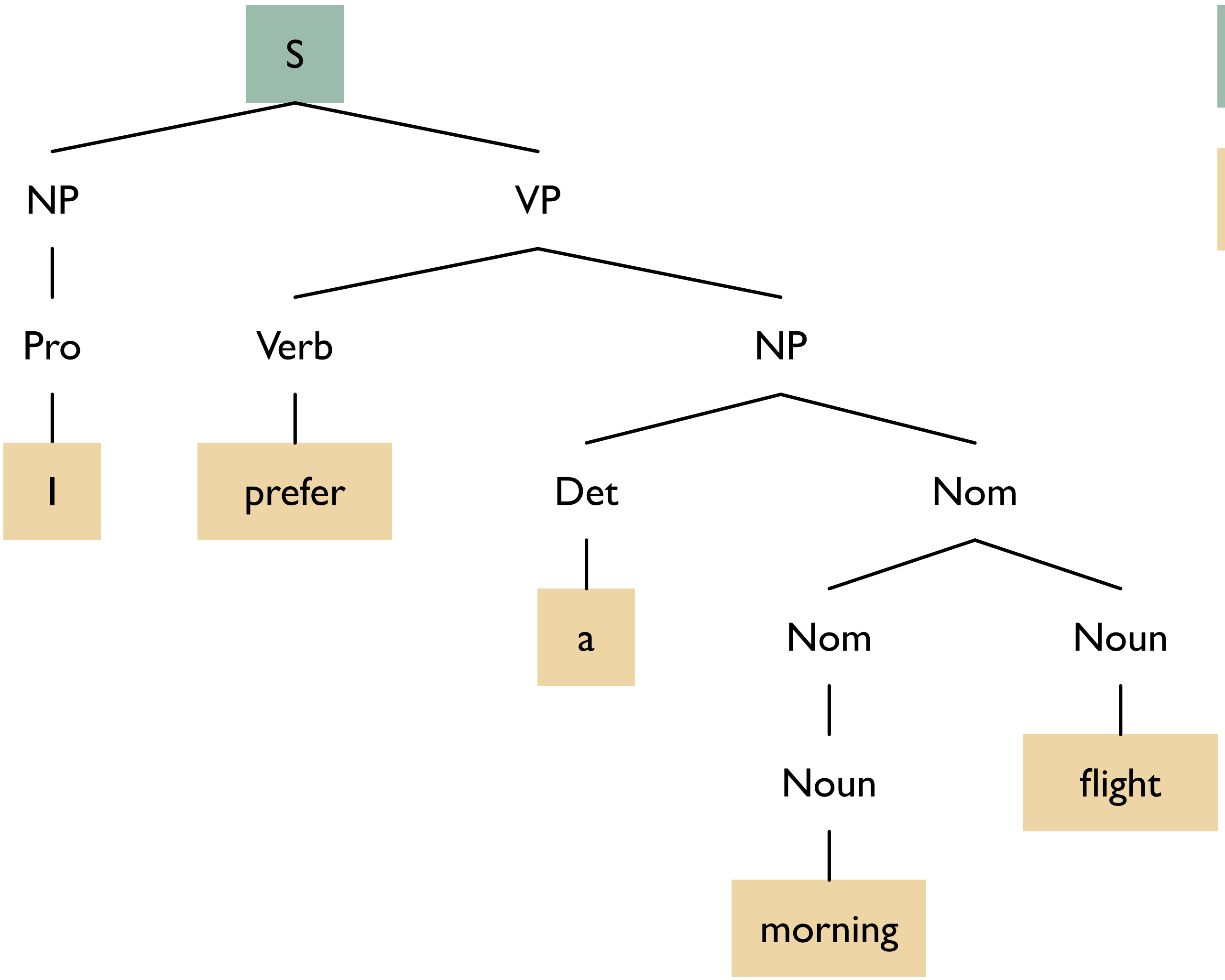


Chomsky

Ajdukiewicz

Tesnière

Phrase structure trees



S

NP VP

Pro Verb NP

I prefer Det Nom

a Nom Noun

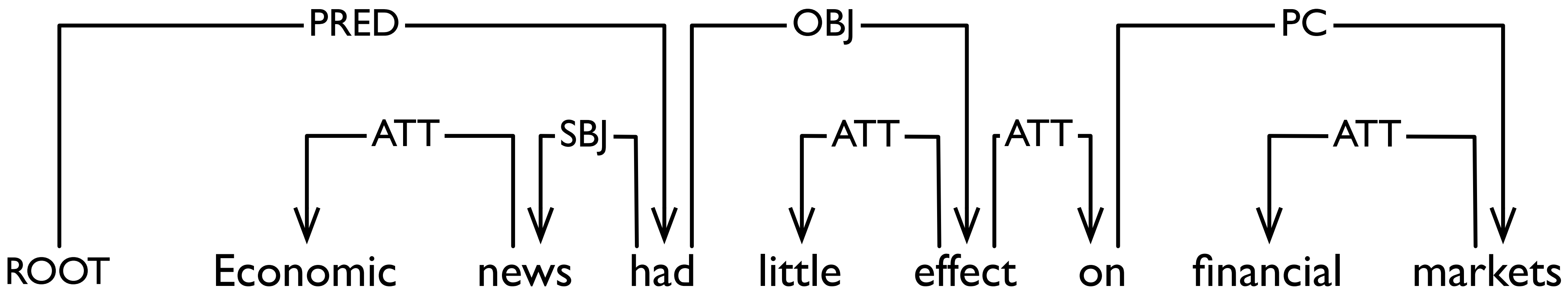
Noun flight

morning

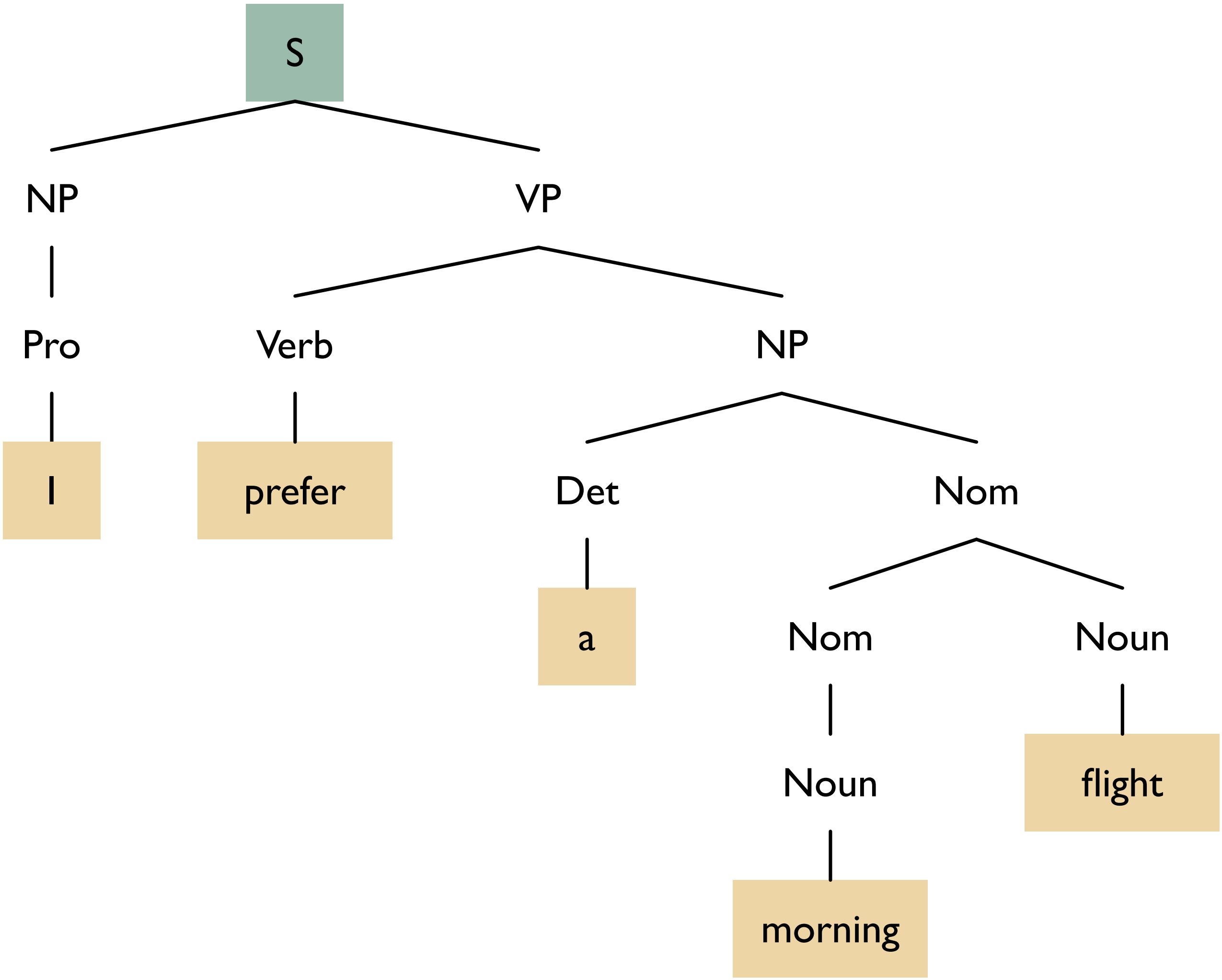
🟩 root (top)

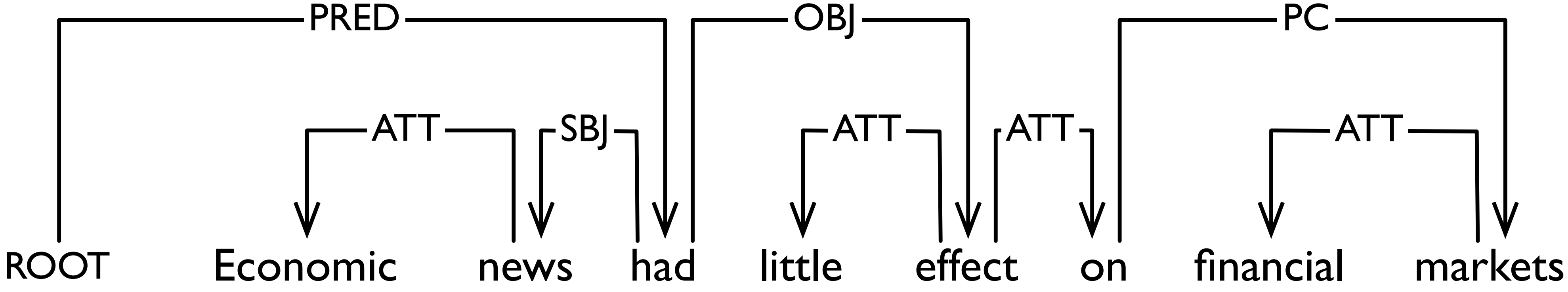
🟨 leaves (bottom)

Dependency trees



Phrase structure vs dependency trees



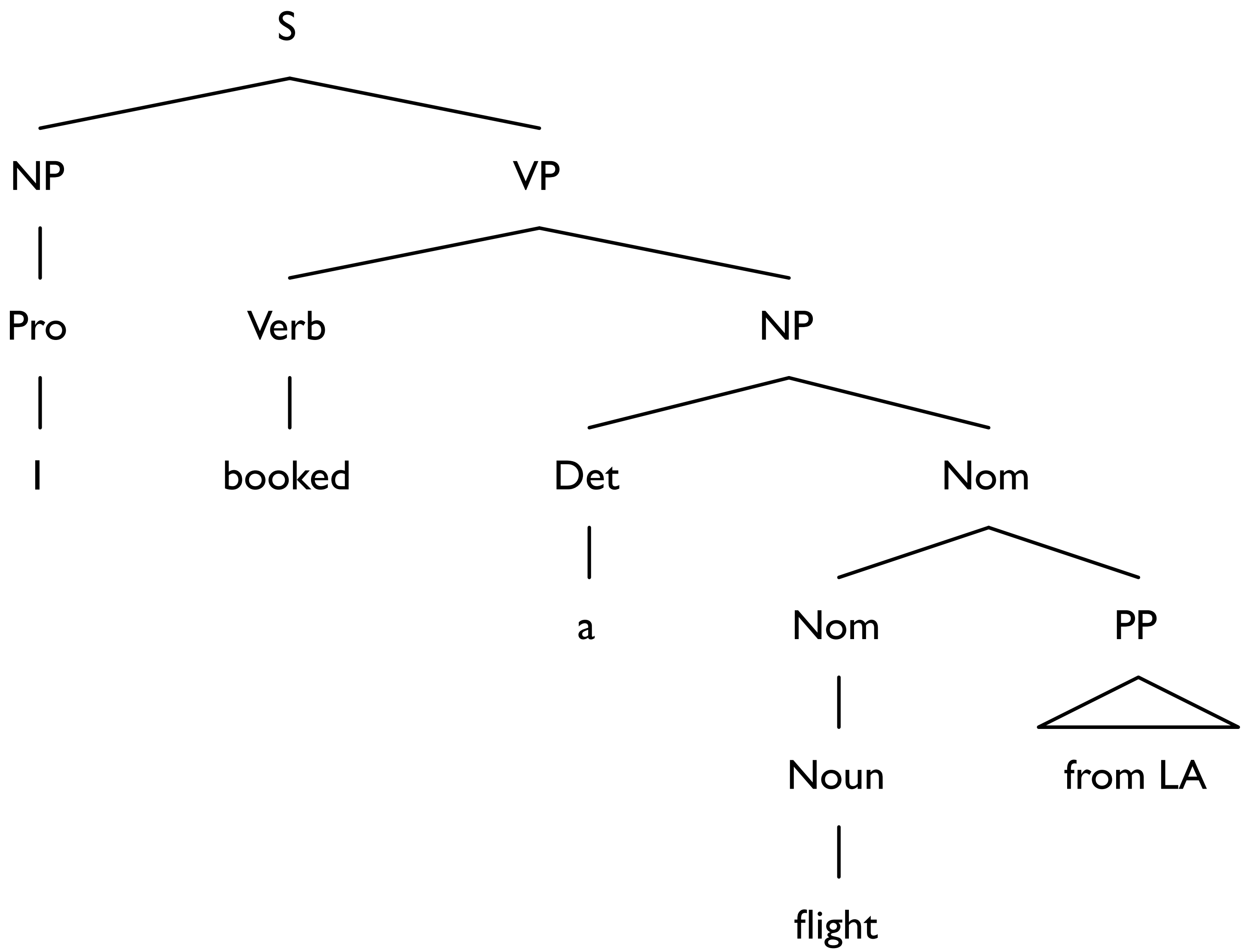


Ambiguity

I booked a flight from LA.

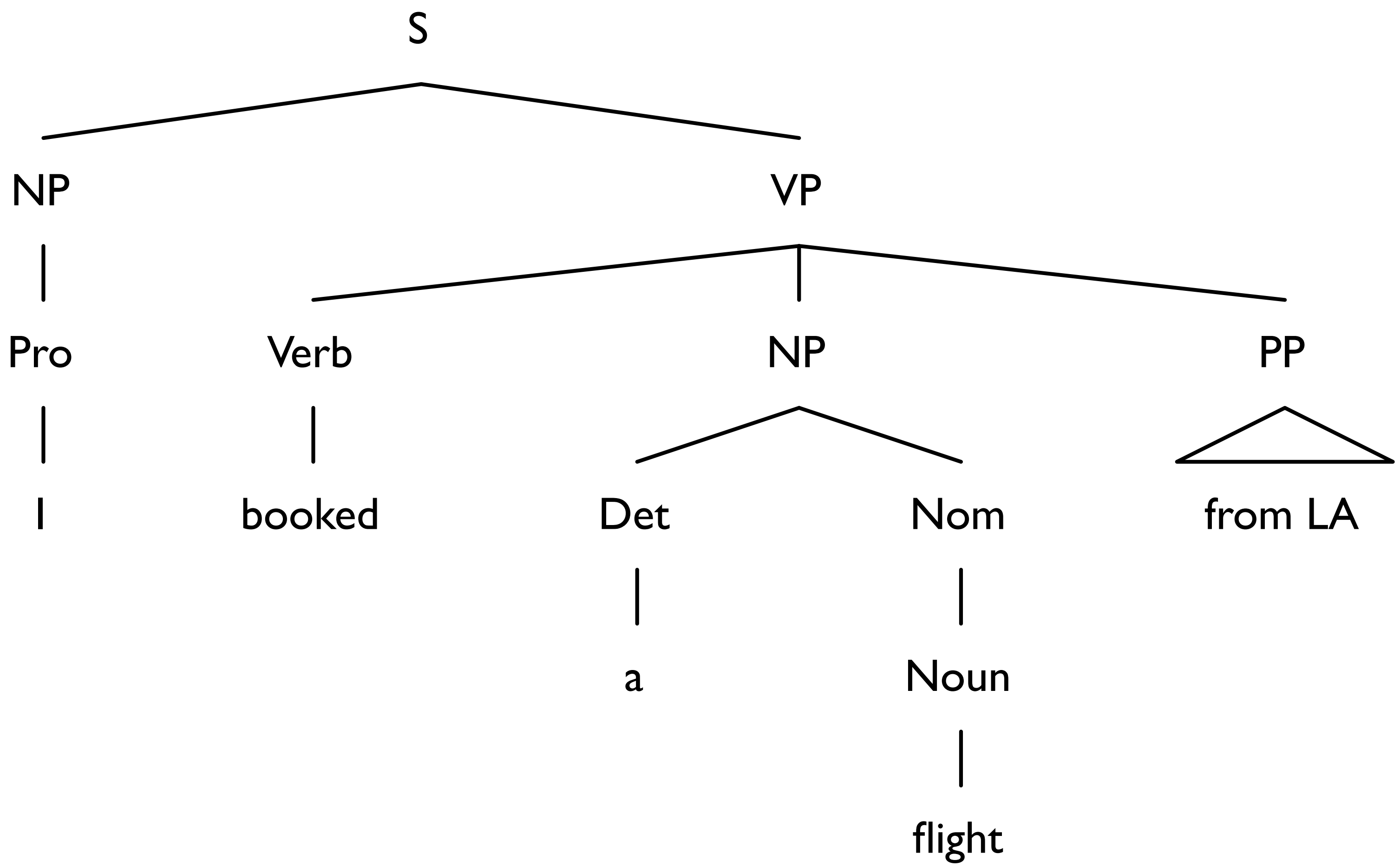
- This sentence is ambiguous. In what way?
- What should happen if we parse the sentence?

Ambiguity



UPPSALA
UNIVERSITET

Ambiguity



Interesting questions

- Is there any parse tree at all?
    - *Recognition*
- What is the best parse tree?
    - *Parsing*

Parsing as search

- Parsing as search:
  Search through all possible parse trees
  for a given sentence.
- In order to search through all parse trees
  we have to ‘build’ them.

Top–down and bottom–up

top–down

only build trees that are rooted at *S*

may produce trees that do not match the input

bottom–up

only build trees that match the input

may produce trees that are not rooted at *S*

Dynamic programming (DP)

- Divide and conquer:
  In order to solve a problem, split it into subproblems, solve each subproblem, and combine the solutions.
- Dynamic programming (DP) (bottom up):
  Solve each subproblem only once and save the solution in order to use it as a partial solution in a larger subproblem.
- Memoisation (top down):
  Solve only the necessary subproblems and store their solutions for reuse in solving other subproblems.

UPPSALA
UNIVERSITET

Example: fibonacci numbers

Naive implementation

```
def fib(n):
    if n <= 1:
        return n
    else:
        return fib(n-1) + fib(n-2)
```

Time complexity: $O(2^n)$

Example: fibonacci numbers

Memoization (top down)

```
fibC = {0:0, 1:1}
def fib_mem(n):
    if n <= 1:
        return n
    if not n in fibC:
        fibC[n] = fib_mem(n-1) + fib_mem(n-2)
    return fibC[n]
```

Time complexity: O(n)

Example: fibonacci numbers

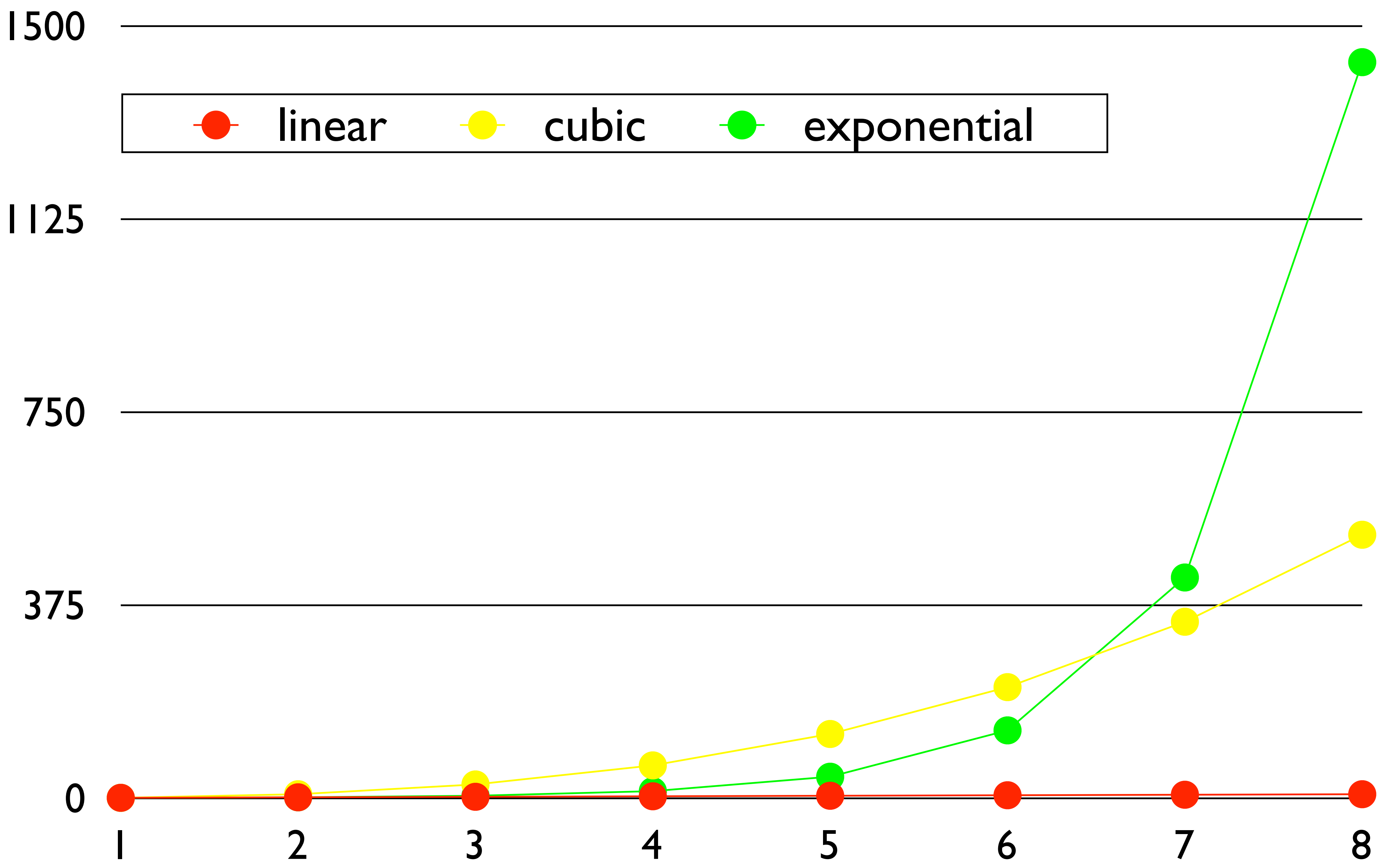
Dynamic programming (bottom up)

```
def fib_dp(n):
    fibV = [0,1]
    for i in range(2, n+1):
        fibV.append(fib[i-1] + fibV[i-2])
    return fibV[n]
```

Time complexity: O(n)

UPPSALA
UNIVERSITET

How many trees are there?



Complexity

- Using DP we can (sometimes) search through all parsetrees in polynomial time.
- That is much better than to spend exponential time!
- But it may still be too expensive!
  In these cases one can use an approximative method such as greedy search or beam search.
  - Often possible in linear time

Course information

Intended learning outcomes 5LN713/5LN717

At the end of the course, you should be able to

- explain the standard models and algorithms used in phrase structure and dependency parsing;
- implement and evaluate some of these techniques;
- critically evaluate scientific publications in the field of syntactic parsing,
- design, evaluate, or theoretically analyse the syntactic component of an NLP system (5LN713)

UPPSALA
UNIVERSITET

Examination 5LN713/5LN717

- Examination is continuous and distributed over four graded assignments, two literature seminars, and a project (for 7,5 credits)
- Two assignments are programming tasks where you implement (parts of) parsers.
- Assignment on evaluating a neural parser
- Literature review assignment
- Two literature seminars

Practical assignments

- Assignment 1: PCFG
  - Implement conversion of treebank to CNF
  - Implement CKY algorithm
- Assignment 3: Dependency parsing
  - Implement an oracle for transition-based dependency parsing
- Assignment 4:
  - Use and evaluate a neural parser: uuparser
  - In a cross-lingual low-resource setting

Literature review

- Pick two research articles about parsing
- Can be from journals, conferences or workshops
- The main topic of the articles should be parsing, and they should be concerned with algorithms (i.e. not focusing on applying parsing to other tasks, evaluation, et.c.)
- Write a 2-page report: summarize, analyse and critically discuss

Literature seminars

- Read one given article for each seminar
- Prepare according to the instructions on the homepage
- Everyone is expected to be able to discuss the article and the questions about it
  - It should be clear that you have read and analysed the article, but it is perfectly fine if you have misunderstood some parts
- The seminars are obligatory
  - If you miss a seminar or are unprepared, you will have to hand in a written report.

Project

- Can be done individually or in pairs:
  - To be self-organized by you!
- Suggestions for topics/themes on web page
- Project activities:
  - Proposal (May 5)
    - Then you will be assigned a supervisor
  - Report (June 5)
  - Oral discussion (only for pairs):

Learning outcomes and examination

- explain the standard models and algorithms used in phrase structure and dependency parsing; all assignments and seminars
- implement and evaluate some of these techniques; assignment 1, 3, 4
- critically evaluate scientific publications in the field of syntactic parsing, assignment 2, seminars
- design, evaluate, or theoretically analyse the syntactic component of an NLP system (5LN713) project

Grading 5LN713/5LN717

- The assignments are graded with G and VG
- G on the seminars if present, prepared and active. The seminars are obligatory!
- To achieve G on the course:
  - G on all assignments, seminars and project
- To achieve VG on the course:
  - VG on at least three assignments
  - VG on project and at least one assignment

Teachers

- Sara Stymne
  - Examiner, course coordinator, lectures, assignments, seminar, project supervision
- Ali Basirat
  - Project supervision

Teaching

- 9-10 lectures
  - Distributed as recordings
  - Followed by discussion sessions in Zoom
- 2 seminars
  - Online in Zoom
- Lab support on Zoom 3 times

Online teaching

- Most interactive activites via Zoom
- Discussion forum in studentportalen
  - Ask questions about course structure, course content, anything there.
  - Share useful contents by your peers
  - Students encouraged to be active!
- Course web page and studentportalen will be used for information

Lectures

- Lectures and course books cover basic parsing algorithms in detail
- They touch on more advanced material, but you will need to read up on that independently

Lecture organization

- Watch recorded lectures (slides+voice) on your own
- Read lecture notes when available
- Read relevant course literature
- Work on given small exercise on your own
- This is followed by a Zoom discussion session
    - Discussion of exercise + questions from recordings
    - Questions by students
    - Potentially repetition of hard-to-grasp concepts!
    - Might be shorter than 2 hours scheduled

Course information

- Web page:
  - Course information
  - Assignments and other instructions
  - Schedule (TimeEdit will not be updated)
- Studentportalen:
  - Zoom links
  - Lectures and lecture materials
  - Discussion forum

Online teaching

- The course was converted to online mode on short notice
- Everything might not work smoothly
    - Let me know!
    - Things can be changed as needed

Course workload 5LN713

- **7.5 hp means about 200 hours work:**
- ~ 40 h lectures (including preparation)
- 2 h seminars
- 158 h work on your own
  - ~ 80 h assignment work (including reading)
  - ~ 10 h seminar preparation
  - ~ 68 h project work (5LN713)

Deadlines

Assignment	Deadline
1: PCFG	April 28
2: Lit review	May 12
3: Dependency	May 22
4: Cross-lingual	May 29
Project proposal	May 5
Project report	June 5
Backup	*August 14*

Seminar	Date
1	April 23
2	May 14

UPPSALA
UNIVERSITET

Reading: course books

- Daniel Jurafsky and James H. Martin.
  Speech and Language Processing. 3rd edition.
  2019. Available online as pdf.
  Chapters 12-14.
- Sandra Kübler, Ryan McDonald,
  and Joakim Nivre. Dependency Parsing.
  Morgan and Claypool, 2009.
  Chapter 1-4, 6.

Reading: articles

- Seminar 1
  - Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, Noah A. Smith. Recurrent Neural Network Grammars. NAACL 2016.
- Seminar 2
  - Eliyahu Kiperwasser and Yoav Goldberg. Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations. TACL. Volume 4, 2016

Reading: additional material

- Lecture notes by Joakim Nivre
- Additional research articles
  - Especially for project and assignment 2

Evaluation from previous years

- Overall score: 2.7 2019 (3.7 in 2018; 4.0 in 2017)
- Good:
  - Explanations of algorithms
  - Programming assignments were useful (but hard)
- Bad:
  - Examination and deadlines were not made clear, and there were too many deadlines
    - It should be clear now! And some tasks are removed.
  - Lab sessions would be helpful
    - This was planned, but now moved to Zoom
  - No practical tasks with actual parsers
    - New assignment added on this.
- The course is largely reverted to pre-2019 format, which has been successful.

Work until Tuesday discussion

- Read J&M 12.1-12.7 (introduction)
- Read J&M 13.1-13.3; 14.1-14.2 (CKY)
- Read lecture notes
- Listen to lectures (once posted)
- Read descriptions of assignments
- Work on exercises
- Repetition (if needed): basic syntax, programming, complexity

Exercise

- Try to come up with parse trees for all possible interpretations of the below example sentence:
  - Phrase-structure trees
  - Dependency trees

- ”Time flies like an arrow”