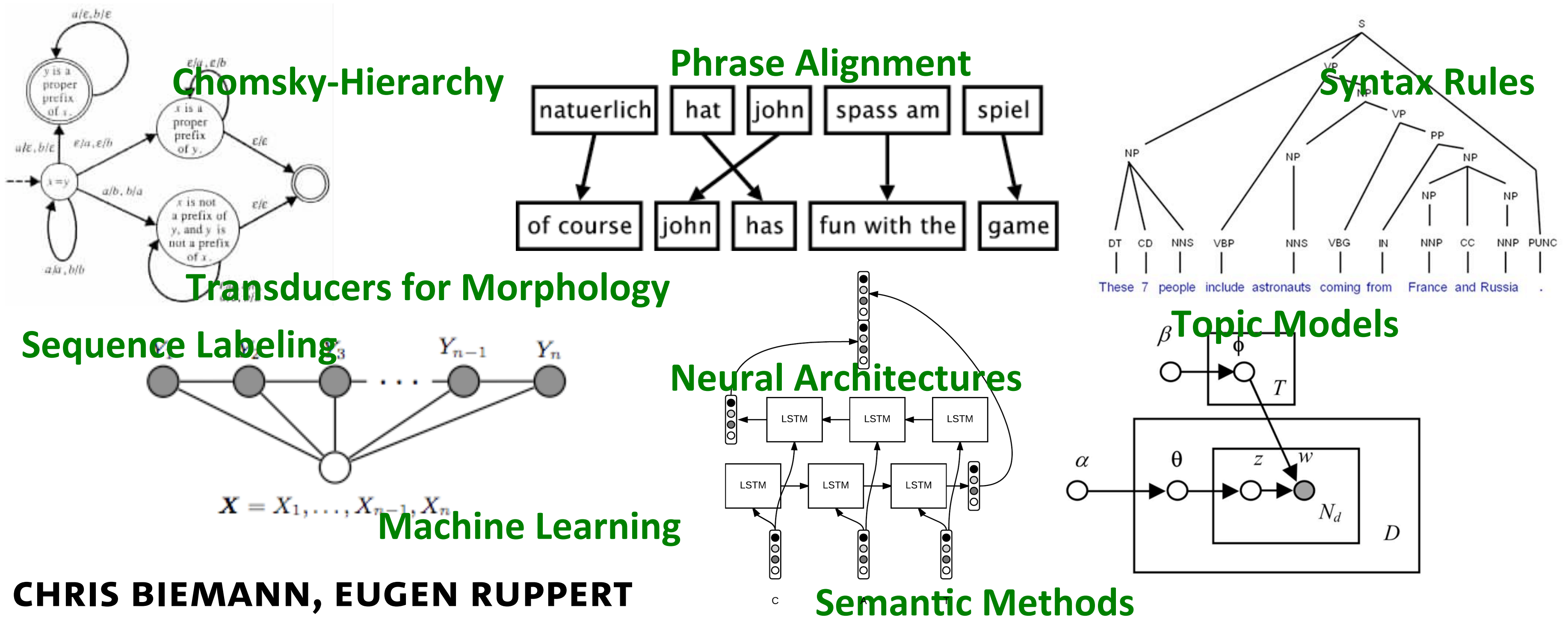
MASTER STUDIES INFORMATICS
UNIVERSITÄT HAMBURG, SS 2019



CHRIS BIEMANN, EUGEN RUPPERT

STATISTICAL METHODS OF LANGUAGE TECHNOLOGY

LECTURERS

Prof. Dr. Chris Biemann
biemann@informatik.uni-hamburg.de

Informatikum Stellingen,
Office: F429
Appointment: by email

Eugen Ruppert, M. Sc.
ruppert@informatik.uni-hamburg.de

Informatikum Stellingen,
Office: G010
Appointment: by email

Dr. des. Seid Muhie Yimam,
yimam@informatik.uni-hamburg.de

Informatikum Stellingen,
Office: F415
Appointment: by email

SHORT INTRO OF THE LT GROUP

HTTP://LT.INFORMATIK.UNI-HAMBURG.DE/

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

- Established in October 2016 at UHH, MIN-Faculty, Comp.Sci. Dept.
- Key Research Areas
  - Statistical Semantics: capture word meaning with big-data methods
  - Structure Discovery: unsupervised-knowledge-free methods for language processing
  - Cognitive Computing: human in the loop, interactive systems
  - Open Source Software and Open Data: committed to the most lenient licenses possible
- Currently: 1 Prof, 4 PostDocs, 6 PhD students, 1 office assistant

LANGUAGE TECHNOLOGY

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

- Formal languages?
- Programming Languages?

Here:

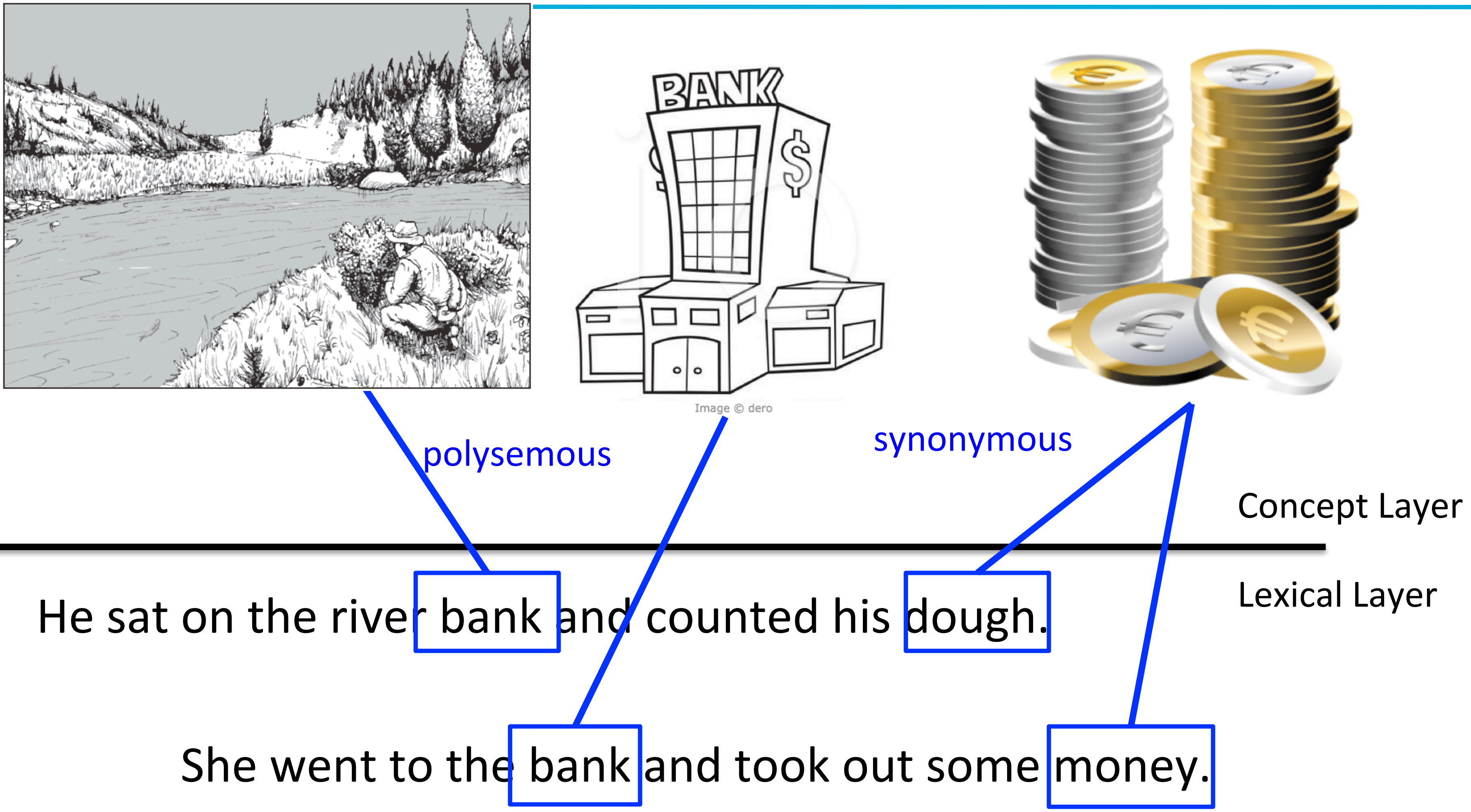
Natural Languages

Natural Language:

- Naturally grown
- Constantly changing
- No well-defined semantics
- Many layers of interpretation
- Meaning dependent on context
- …

Technologies coping with this

WHY LANGUAGE IS HARD ..



He sat on the river bank and counted his dough.

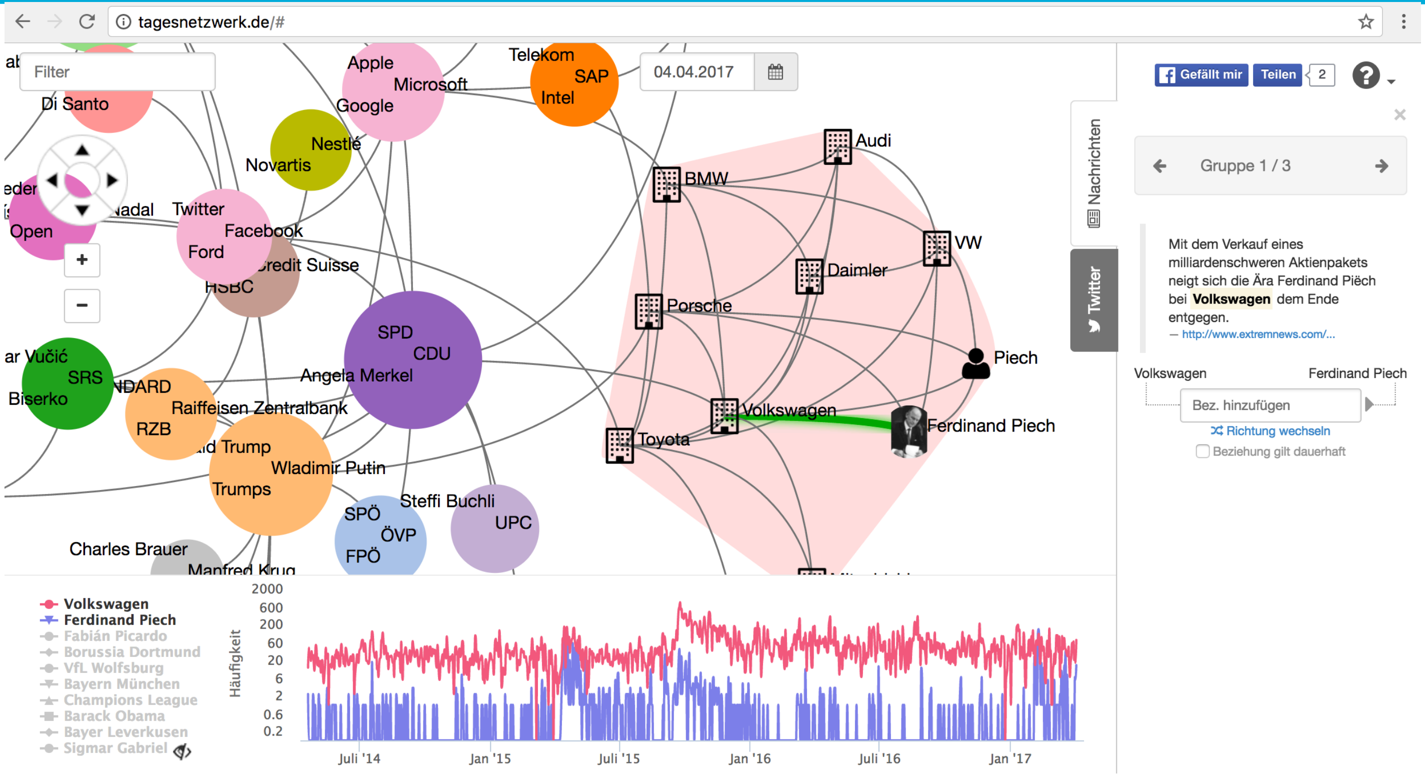
She went to the bank and took out some money.

RECENT PROJECTS OF LANGUAGE TECHNOLOGY GROUP

- BMBF: WebAnno: A web-based annotation platform
- DFG: Joining Ontologies and Distributional Semantics
- DFG: Semantic Writing Aid
- DB: Aspect-based Sentiment Analysis
- LFF: Forum 4.0: Analysis of User Comments
- IBM: JoBimText: Linking text to Knowledge with Distributional Semantics
- Telekom: Xpert Finder
- VW: New/s/leak: Network of Searchable Leaks

NETZWERK DES TAGES
WWW.TAGESNETZWERK.DE

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG



Mit dem Verkauf eines milliardenschweren Aktienpakets neigt sich die Ära Ferdinand Piëch bei **Volkswagen** dem Ende entgegen.
— http://www.extremnews.com/...

- Student project, up since 2014!

OVERVIEW OF THIS LECTURE

Statistical Methods of Language Technology

- focus on methods rather than applications
- variety of techniques, focus on statistical methods
- efficiency vs. effectiveness

Statistical Methods of **Language Technology**

- cores of methods being used in NL systems
- adaptations of generally known algorithms to language data
- evaluation of techniques

Lecture: theory, concepts, algorithms
Practice class: hands-on, writing small programs, using available software

TEXTBOOKS

- Jurafsky, D. and Martin, J. H. (2009): **Speech and Language Processing. An Introduction to Natural Language Processing**, Computational Linguistics and Speech Recognition. Second Edition. Pearson: New Jersey

- Manning, C. D. and Schütze, H. (1999): **Foundations of Statistical Natural Language Processing**. MIT Press: Cambridge, Massachusetts

- Carstensen, K.U., Ebert, Ch., Endriss, C., Jekat, S., Klabunde, R. and Langer, H. (Editors) (2004): **Computerlinguistik und Sprachtechnologie. Eine Einführung**. 2. Auflage. Spektrum: Heidelberg

Literature for specialized topics will be given in-place.

LEARNING GOALS

- understand statistical methods for language processing in detail
- feeling for language tech applications, avoiding pitfalls
- ability to plan technology requirements for a language tech project
- analyze and evaluate the use of NLP in applications
- see the beauty of language technology, be ready to write your thesis in language tech

PRACTICE CLASS INFORMATION

- In the practice classes you will work on weekly assignments, which will give you some practical experience in NLP
- The assignments will be graded on a binary scale (“ok”/”not ok”)
- You need 50% of points to pass the course
- The practice classes are supervised by Eugen Ruppert and Seid Muhie Yimam
- Depending on nature of the topic, assignments will be
  - theoretical, i.e. paper-and-pencil
  - practical, i.e. writing a program and applying it to data
  - hands-on, i.e. applying a third-party program to data

ORGANISATIONAL INFORMATION

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

- The lecture slides, handouts, readings etc. can all be found on the Moodle e-Learning platform: https://lernen.min.uni-hamburg.de/moodle/course/view.php?id=27
- The required enrolment key is: **SMOLT2019**
- We use the *Participant's Forum* there for discussion and Q&A (encouraged!)

FINAL EXAM

- How: Written exam 2h

- When:
  - 25. Jul. 2019 09:30-11:30 ESA C
  - 17. Sep. 2019 09:30-11:30 Hörs Pharmazie (gr)

- Content:
  - Lecture
  - Exercises
  - Reading

TIME SLOTS FOR CLASSES 2018

time\day	MONDAY	TUESDAY	WEDNESDAY
11-12			
12-13			12:15-13:45 F334 Lecture
13-14			
14-15			14:15 – 15:45 F334 Practice Class I
15-16			
16-17	16:15 – 17:45 G 203 Practice Class II		
17-18			
18-19			

TOPICS OF THIS LECTURE

- Formal Languages and Automata
- Computational Morphology
- Sequence Tagging
- Topic Modelling
- Statistical Machine Translation
- Graph-Based Methods
- Distributional Semantics
- Word Senses and their Disambiguation

- Chomsky, Noam (1959). "On certain formal properties of grammars". Information and Control 2 (2): 137–167
- Klabunde, R. (2004): Automatentheorie und Formale Sprachen. In: Carstensen, K.U., Ebert, Ch., Endriss, C., Jekat, S., Klabunde, R. and Langer, H. (Editors): Computerlinguistik und Sprachtechnologie. Eine Einführung. 2. Auflage. Spektrum: Heidelberg
- Hopcroft, J.E., Motwani, R. and Ullman, J.D. (2006): Introduction to Automata Theory, Languages, and Computation. Third Edition. Pearson: New Jersey

FORMAL LANGUAGES AND AUTOMATA

CHOMSKY HIERARCHY OF FORMAL LANGUAGES

RECAP: FORMAL LANGUAGES AND AUTOMATA

- Automata theory and theory of formal languages are a part of theoretical computer science
- Their concepts originate in theoretical linguistics: Noam Chomsky is the originator of the Chomsky hierarchy of formal languages

Why talk about it?

- complexity of sub-systems of natural language informs complexity of automatic processing machinery
- Fundamental results from theoretical computer science have direct implications on implementations for language technology applications

DEFINITIONS

- A **LANGUAGE** is a collection of sentences of finite length all constructed from a finite alphabet of symbols

- A **GRAMMAR** can be regarded as a device that enumerates the sentence of a language

- A grammar of language L can be regarded as a function whose range is exactly L

FORMAL GRAMMAR

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

A **formal grammar** is a quad-tuple *$G = (\Phi,\Sigma,R,S)$* where

- Φ is a finite set of non-terminals
- Σ a finite set of terminals, disjoint from Φ
- R a finite set of production rules of the form
  $\alpha \in (\Phi \cup \Sigma)^* \rightarrow \beta \in (\Phi \cup \Sigma)^*$ with $\alpha \neq \varepsilon$ and $\alpha \notin \Sigma^*$
- S, Element of Φ : start symbol

DERIVATION, FORMAL LANGUAGE, AUTOMATON

Let $G = (\Phi,\Sigma,R,S)$ be a formal grammar and let $u,v \in (\Phi \cup \Sigma)^*$.

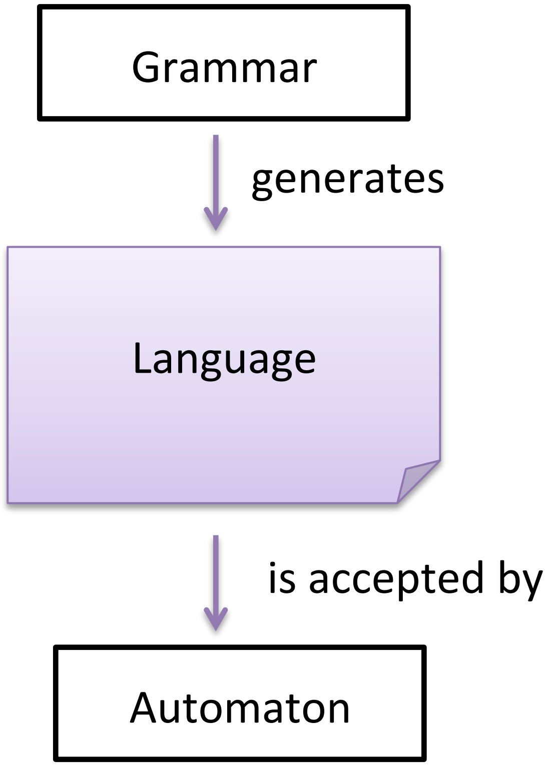
1. v is **directly derivable** from u, noted $u \Rightarrow v$, if
   $u = awb$, $v=azb$ and $w \rightarrow z$ is a production rule in R.

2. v is **derivable** from u, noted $u \overset{*}{\Rightarrow} v$, if there are words $w_0..w_k$, such that $u \Rightarrow w_0$, $w_{n-1} \Rightarrow w_n$ for all $0<n\leq k$ and $w_n \Rightarrow v$.

Let $G = (\Phi,\Sigma,R,S)$ be a formal grammar. Then, $L(G) = \{w \in \Sigma^* \mid S \overset{*}{\Rightarrow} w\}$ is the **formal language** generated by G.

An **automaton** is a device that decides, whether a given sentence belongs to a formal language.

GENERATION AND ACCEPTANCE

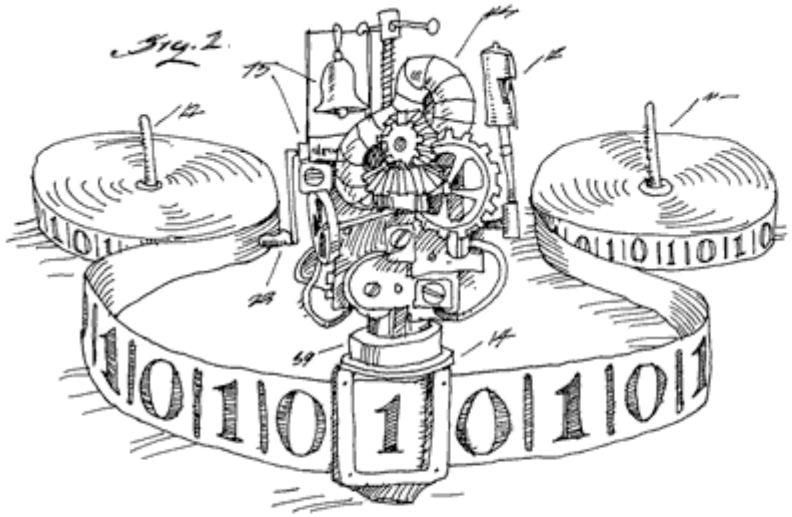


The complexity of the generating grammar influences the complexity of the accepting automaton

TYPE-0 GRAMMAR: UNRESTRICTED

The mechanism of unrestricted grammars allows the definition of very complex languages that in turn need very complex automata. Restrictions on the form of production rules lead to different types of grammars.

An **unrestricted** formal grammar is called **Type-0 grammar** and can be accepted by a **Turing Machine**.
It produces **recursively enumerable** languages.



TYPE-1-GRAMMAR: CONTEXT-SENSITIVE

A grammar *$G = (\Phi,\Sigma,R,S)$* is **context-sensitive**, iff all production rules in R obey the form

- either $\alpha A\gamma \rightarrow \alpha\beta\gamma$ with $\alpha,\beta,\gamma \in (\Phi \cup \Sigma)^*$, $A \in \Phi$, $\beta \neq \varepsilon$
- or $S \rightarrow \varepsilon$.

If $S \rightarrow \varepsilon$, then S cannot appear in the righthand side of rules in R.

The language of a type-1 grammar is accepted by a **linear bounded automaton** (a nondeterministic Turing machine whose tape is bounded by a constant times the length of the input).

TYPE-2 GRAMMAR: CONTEXT-FREE

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

A grammar $G = (\Phi,\Sigma,R,S)$ is **context-free**, iff all production rules in *R* obey the form $A \rightarrow \alpha$ with $A \in \Phi$, $\alpha \in (\Phi \cup \Sigma)^*$.

Context free grammars are also called **phrase structure** grammars.

Context-free languages are closed under the following operations:

- Union: if *F* and *G* are context-free, so is $F \cup G$
- Concatenation: if *F* and *G* are context-free, so is $F \bullet G$
- Kleene Star: if *G* is context-free, so is G^*

Context-free languages are not closed under the following operations:

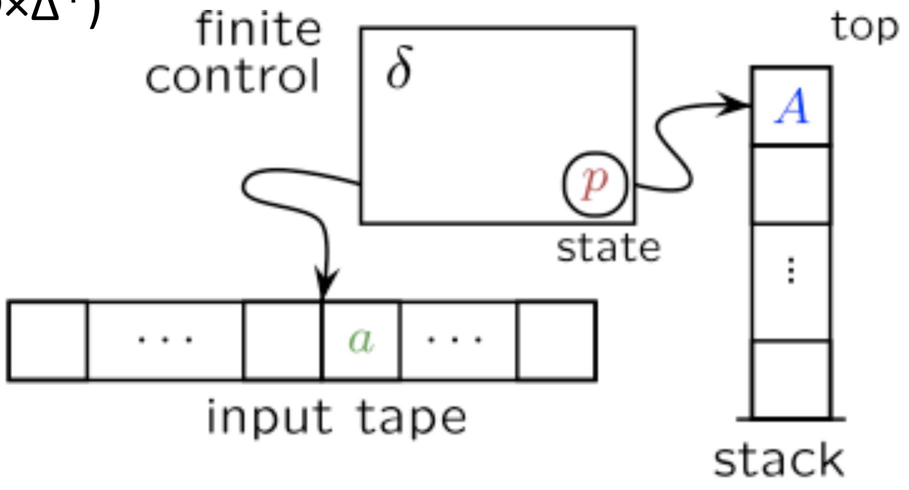
- Intersection: from *F* and *G* are context-free does not follow that $F \cap G$ *is.*
- Complement: from *G* context-free does not follow that $\neg G$ is.
- Difference: from *F* and *G* are context-free does not follow that $F \setminus G$ *is.*

PUSHDOWN AUTOMATON

Context-free grammars are accepted by **non-deterministic pushdown automata**.

A non-deterministic push-down automaton PDA=(*Φ,Σ,Δ,□,δ,S,F*) consists of

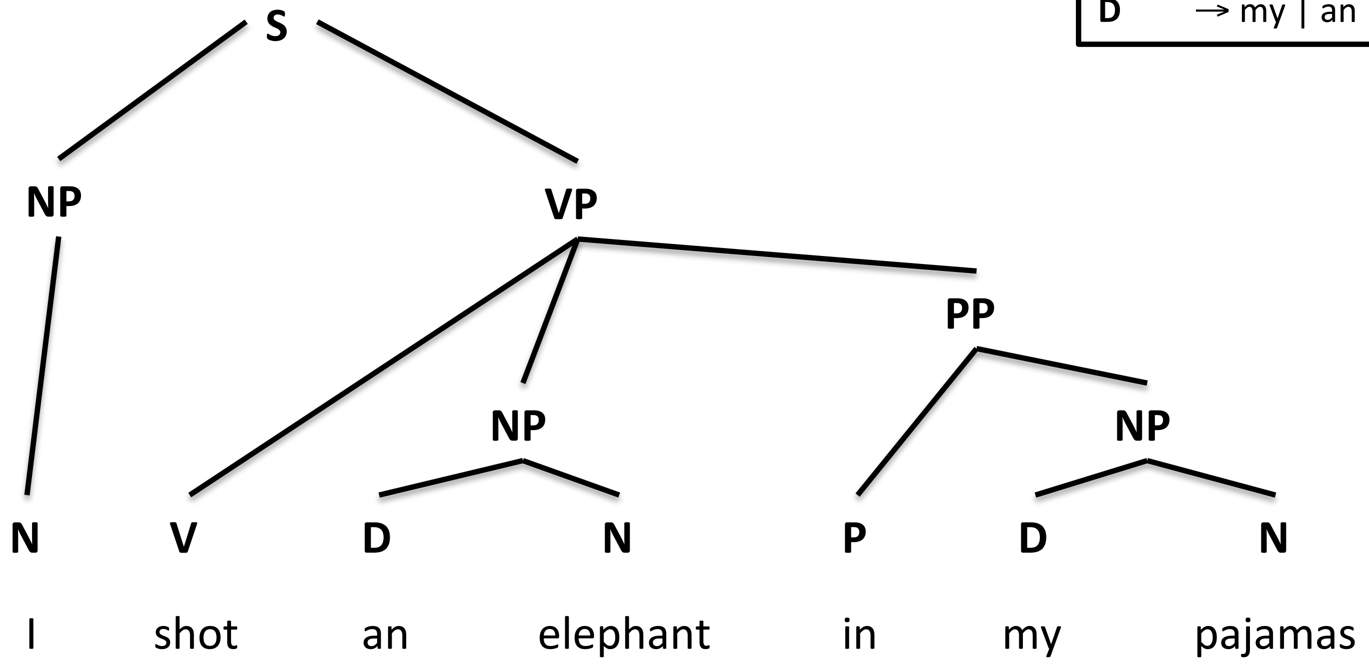
- Alphabet Φ of states
- Alphabet Σ of input symbols, disjunct with Φ
- Alphabet Δ of stack symbols, disjunct with Φ
- initial stack symbol □
- transition relaton δ: Φ×Σ×(Δ∪□) →ρ(Φ×Δ*)
- start state S ∈ Φ
- set of final states F⊂Φ



CONTEXT-FREE SYNTAX TREES WITH PHRASE STRUCTURE GRAMMARS

- Syntax trees can (almost) be modeled with context-free languages

S	**→ NP VP**
NP	**→ N | D N | NP PP**
VP	**→ V NP | V NP PP**
PP	**→ P NP**
N	→ I | elephant | pajamas
V	→ shot
P	→ in
D	→ my | an | a | the



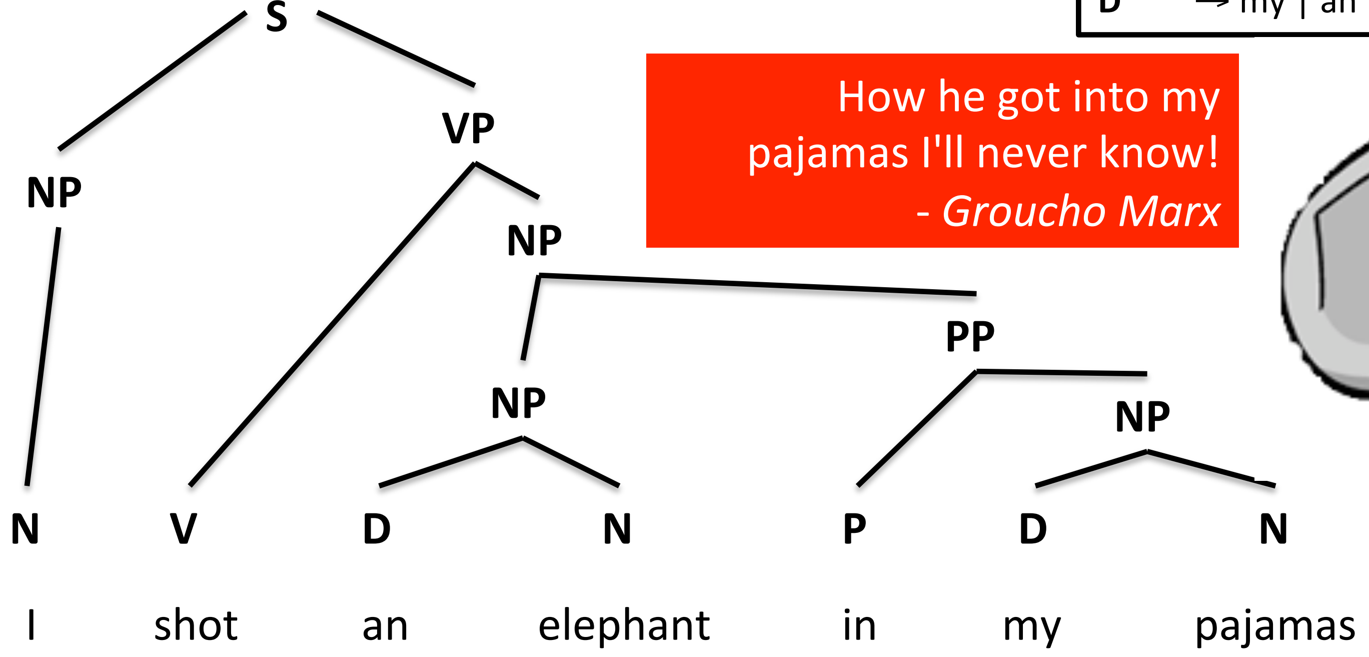
CONTEXT-FREE SYNTAX TREES WITH PHRASE STRUCTURE GRAMMARS

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

- Syntax trees can (almost) be modeled with context-free languages
- one surface sentence can have several derivations

S	**→ NP VP**
NP	**→ N | D N | NP PP**
VP	**→ V NP | V NP PP**
PP	**→ P NP**
N	→ I | elephant | pajamas
V	→ shot
P	→ in
D	→ my | an | a | the

How he got into my pajamas I'll never know!
- Groucho Marx



TYPE-3 GRAMMAR: REGULAR, LEFT/RIGHT LINEAR

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

A grammar *G = (Φ,Σ,R,S)* is **right linear (left linear)**, iff all the production rules in *R* obey the forms

- $A \rightarrow w$
- $A \rightarrow wB$ (left linear: $A \rightarrow Bw$)

with $A,B \in \Phi$ and $w \in \Sigma^*$.

Left (right) linear grammars generate **regular** languages:

- $\varnothing$ is regular
- $\{a_i\}$ is regular for $a_i \in$ alphabet Σ
- if the sets L_1 and L_2 are regular, then $(L_1 \cap L_2)$ is regular
- if the sets L_1 and L_2 are regular, then $(L_1 \bullet L_2)$ is regular
- if set L is regular, then also L^*.

These languages can be described through **regular expressions**.

FINITE STATE AUTOMATON

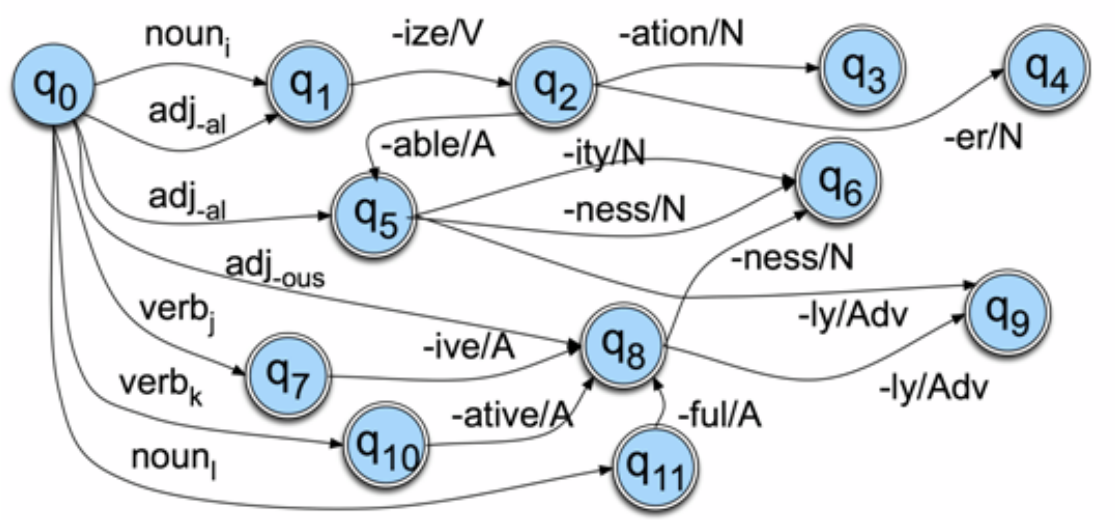
Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Regular grammars are accepted by finite state automata.

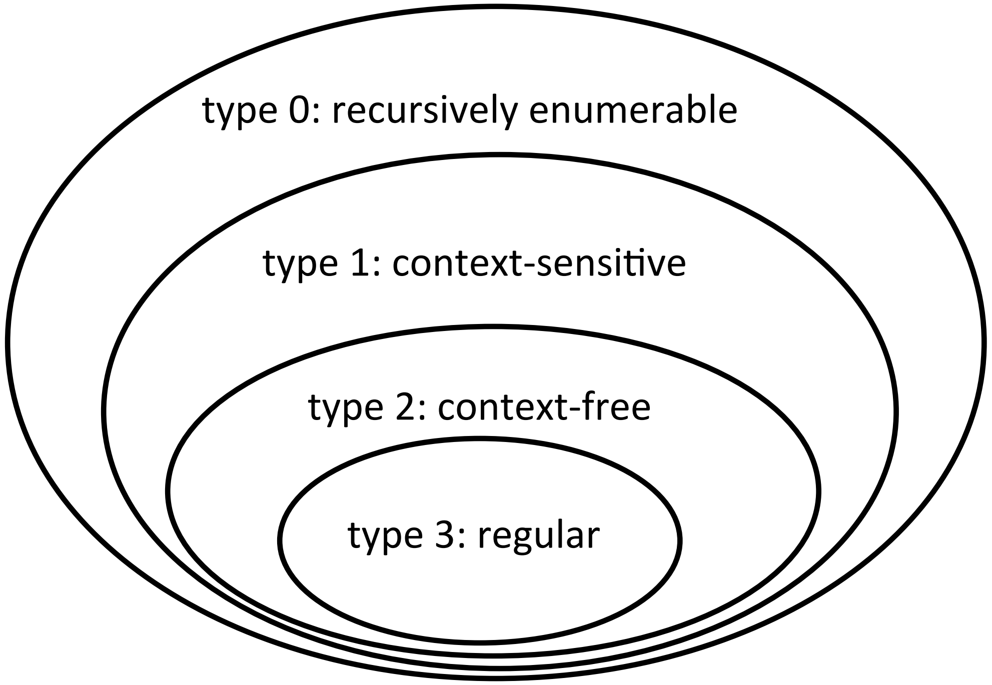
A (deterministic) **finite state automaton** FSA=(Φ,Σ,δ,S,F) consists of:

- set of states Φ
- input alphabet Σ, disjunct with Φ
- transition function δ: $\Phi \times \Sigma \rightarrow \Phi$
- one start state $S \in \Phi$
- set of final states $F \subset \Phi$

Regular languages cover sub-systems of language, such as morphology and chunk parsing.



THE CHOMSKY HIERARCHY OF FORMAL LANGUAGES



Turing Machine (TM)

linearly bounded TM

Pushdown Automaton (PDA)

Finite State Automaton (FSA)

- The different classes are proper subsets of each other: the expressivity of type-(n) grammars is truly smaller than type-(n-1) grammars.
- Several other classes are known, e.g. corresponding to deterministic context-free grammars, tree adjoining grammars …

PROGRAMMING LANGUAGE VS. NATURAL LANGUAGE

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Grammar of Programming languages

- by design: deterministic context free (in most cases)
- which allows efficient parsing
- without ambiguities.
- clearly defined semantics

Grammar of Natural Languages

- somewhere between type-1 and type-2
- many possible parses for a single sentence
- inherent ambiguities
- semantics yet another layer

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

- Jurafsky, D. and Martin, J. H. (2009): Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition. Second Edition. Pearson: New Jersey: Chapter 3.2
- Carstensen, K.U., Ebert, Ch., Endriss, C., Jekat, S., Klabunde, R. and Lange . (Editors) (2004): Computerlinguistik und Sprachtechno Eine Einführu Aufla ktrum: Heidelberg, pages 198-205
- G. Heyer, U. Quasthoff, T. Wittig (Eds.) (2006): apitel 3.4

coming up next

Transducers, Compact Patricia Tries nd D Gs

FINITE STATE MORPHOLOGY