

CRISTIANO CHESI
NETS, IUSS LABORATORY
FOR NEUROLINGUISTICS,
COMPUTATIONAL LINGUISTICS
AND THEORETICAL SYNTAX

Introduction to Linguistic Computation & Complexity Theory

Ph.D. in Theoretical and Experimental Linguistics (TEL)



Index

- ◉ **Formal Grammars**
 - ◉ Formal approaches to **Linguistic Competence**
 - ◉ Phrase Structure Grammars (**PSG**) and **Chomsky's hierarchy**
- ◉ **A theory for (linguistic) computation**
 - ◉ Linguistic formalisms in **processing**
 - ◉ Introduction to **computability** and **complexity**.
 - ◉ **Parsing** as a prerequisite for **understanding**
- ◉ **Advances in linguistic formalisms and processing**
 - ◉ **Competence & Performance**
 - ◉ **Minimalist** Derivations
 - ◉ Complexity as **intervention**
 - ◉ **Small Language Modeling** and **evaluation**

Essential References

Introductory

- ◉ Jurafsky, D. & Martin, J. H. (2009) **Speech and Language Processing**. Prentice-Hall (II edition!).
<http://www.cs.colorado.edu/~martin/slp.html>
(Only chapters 2, 12 and 13)
- ◉ Partee B., A. ter Meulen & R. Wall (1990) **Mathematical Methods in Linguistics**. Springer, 1990
(Only chapters 16 – 18)

Advanced

- ◉ Chesi (2025) **Is it the end of (generative) linguistics as we know it?** IJoL (<https://www.italian-journal-linguistics.com/2025-2/>)
- ◉ Chesi C. (2021) **Expectation-based Minimalist Grammars**
<https://lingbuzz.net/lingbuzz/006135>
- ◉ Chesi, C., & Canal, P. (2019). **Person features and lexical restrictions in Italian clefts**. *Frontiers in Psychology*, 10, 2105.
<http://dx.doi.org/10.3389/fpsyg.2019.02105>
- ◉ Van Dyke, J. A., & McElree, B. (2006) **Retrieval interference in sentence comprehension**. *Journal of Memory and Language*, 55(2), 157-166.

Extended References

- ⦿ Baddeley, A. (2013)
Essentials of human memory (classic edition). Psychology Press.
- ⦿ Chesi C., A. Moro (2014)
Computational complexity in the brain. in Frederick J. Newmeyer and Laurel B. Preston (eds.), *Measuring Linguistic Complexity*. Oxford: OUP
- ⦿ Hopcroft, Motwani & Ullman (2001)
Introduction to the automata theory, languages and computation. Addison-Wesley. Boston
- ⦿ Stabler, E. 1997.
Derivational minimalism. in Retoré, ed. *Logical Aspects of Computational Linguistics*. Springer
- ⦿ Sprouse, J., Wagers, M., & Phillips, C. (2012).
Working-memory capacity and island effects: A reminder of the issues and the facts. *Language*, 88



STAGE I

Formal grammars

(the shopping list
to prepare a
linguistic cake)



Plato's problem



How is it possible that human beings, whose **contacts with the world** are **brief, personal, and limited**, are still able to know so much?



⦿ A **linguistic** example:

⦿ **Which** car do you think the mechanic fixed _ ?

⦿ ***Which** car do you think the mechanic fixed **the engine** ?



⦿ A **logical** conclusion:

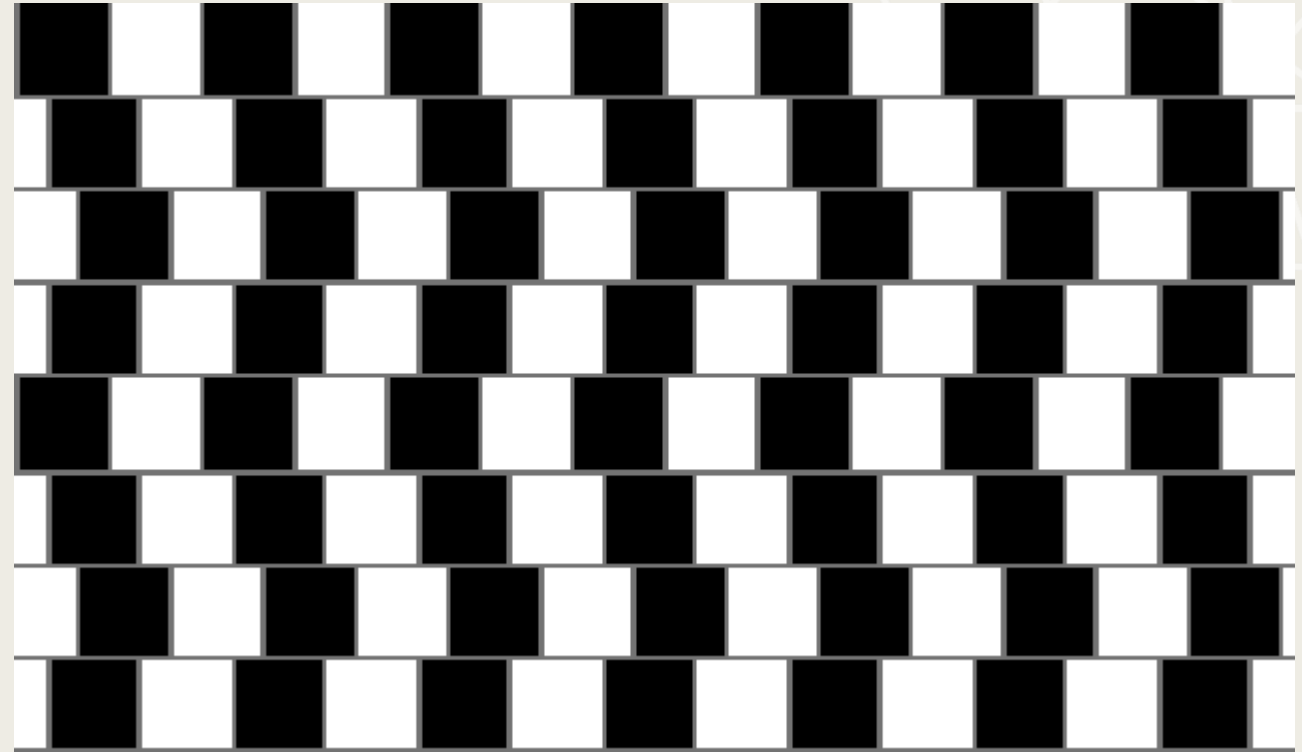
We come with an **innate predisposition** to assign a **precise structural analysis** to the (linguistic) **input** we receive (and **not infinite other possible analyses**)

- ⦿ How is it possible that of **all the information we receive** as input **we manage to process so little** of it and in such a **distorted way**?
- ⦿ A **Linguistic example**:
 - ⦿ The red car the coachbuilder took to the garage had a perforated fender, a dented wheel, and a scratch on the upholstery.
 - ⦿ The mechanic repaired a car that had a punctured wheel, a dented fender and a scratch on the windshield.
- ⦿ A **logical conclusion**:
 - ⦿ Our **processing** of an input (linguistic) **neglects numerous details** that are **reconstructed** by relying on our (linguistic) **expertise** and acquired knowledge of the world (**bias**)

Orwell's problem



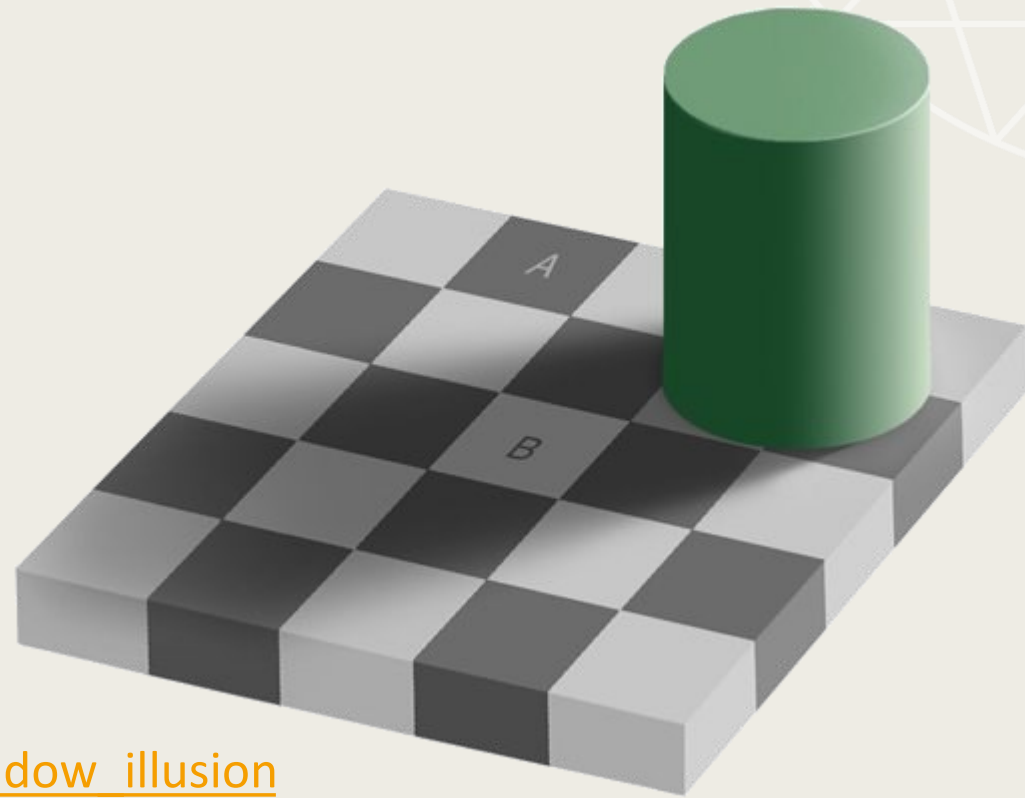
What you see is not what is there



Münsterberg figure

https://en.wikipedia.org/wiki/Caf%C3%A9_wall_illusion

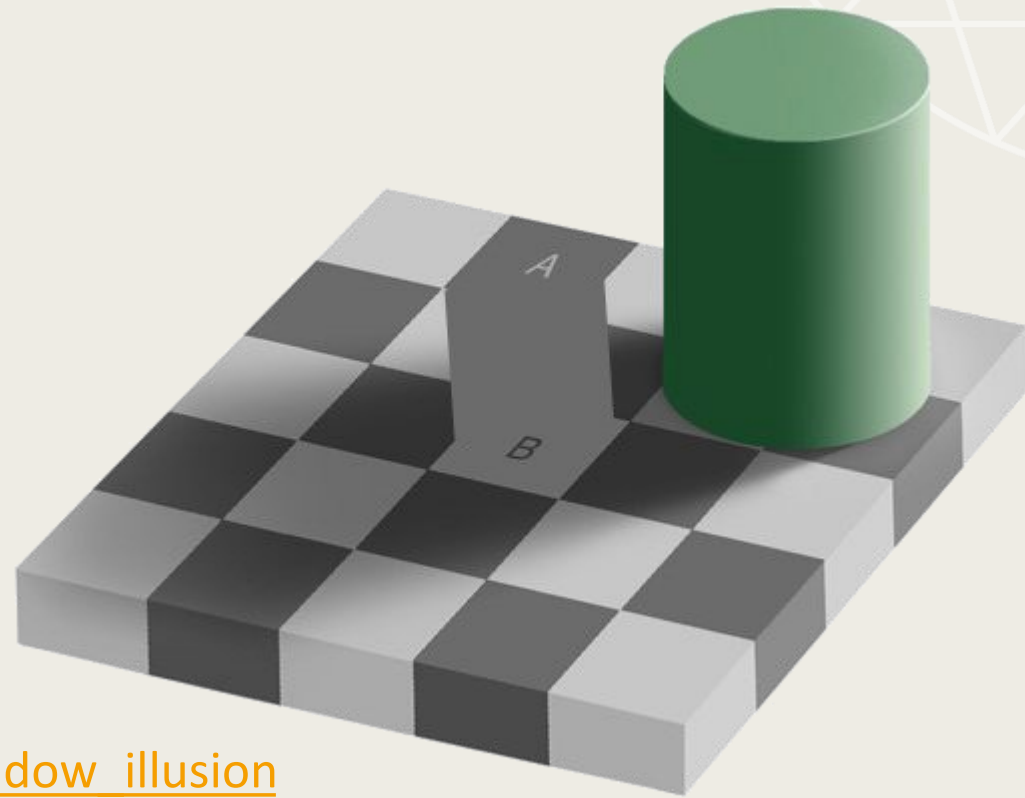
What you see is not what is there



Adelson figure

https://en.wikipedia.org/wiki/Checker_shadow_illusion

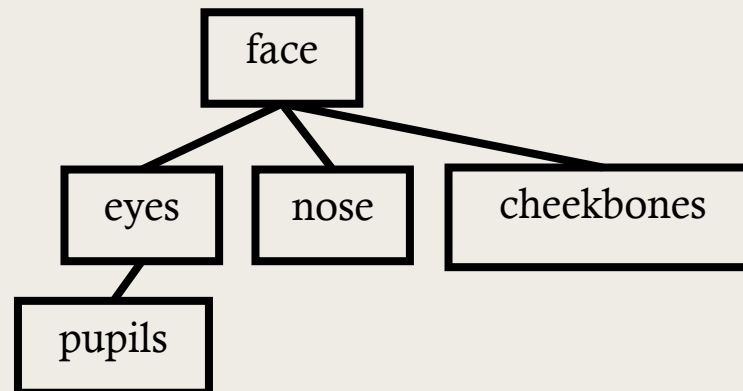
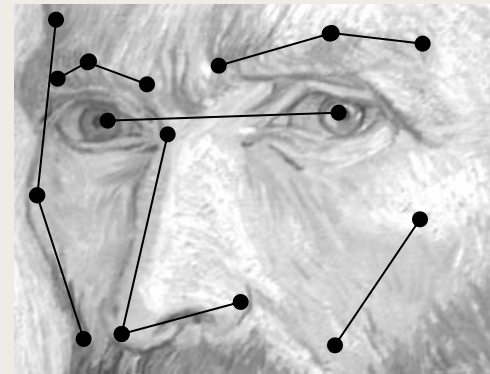
What you see is not what is there



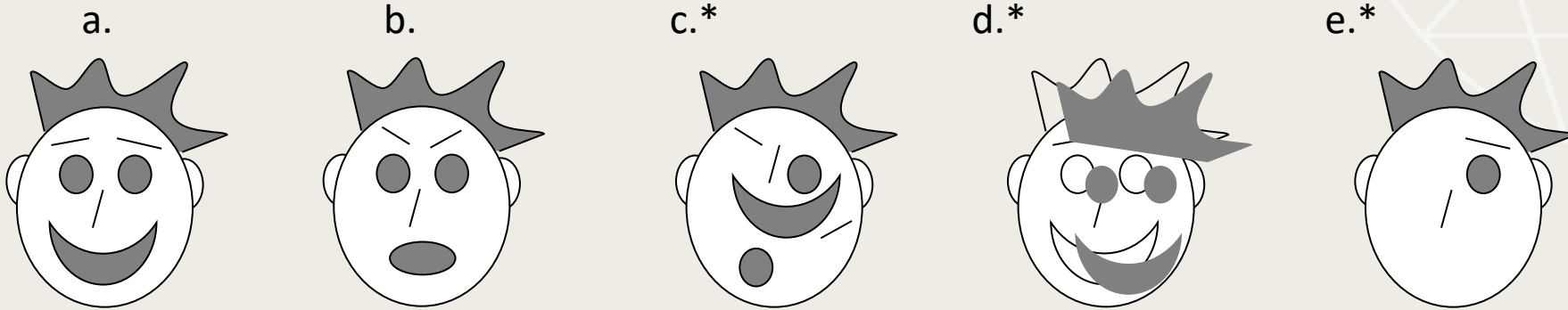
Adelson figure

https://en.wikipedia.org/wiki/Checker_shadow_illusion

A structural approach to Vision



A structural approach to Vision



Il bambino
chiude la porta

Il bambino
apre la porta

*Bambino il
porta chiude la

*I bambino
chiudono le porta

*Il bambino
chiude

The child
closes the door

The child
opens the door

*Child the
closes door the

*The childs
closes the door

*The child
closes

Linguistic Competence

⦿ What kind of **competence** (information structure) do we have?

- ⦿ A word can start by *wo...* (*word*) but not by *wb...*
- ⦿ The *s* in “sings” is different from the one in “roses”
- ⦿ “the rose is beautiful” Vs. *“the is beautiful rose”
- ⦿ “The cat chases the dog” >
subj: cat(**agent**); verb: chase(**action**); obj: dog(**patient**)
- ⦿ ?the television chases the cat
- ⦿ “the houses” Vs. “some house”

⦿ **Linguistic competence** is a **finite** knowledge that allows us to:

- ⦿ Recognizing as grammatical **an infinite** set of expressions
- ⦿ Assigning to them the correct meaning(s)

Linguistic Competence

Simple properties:

- Word order > meaning ... but consider **structural ambiguities!**

e.g. I saw a man in the park with a binocular

- Agreement

e.g. *la mela rosso (lit. *the.F red.F/*M apple.F*)
Gianni ha visto Maria vs. Gianni l'ha vista
G.M has seen.M/*F M.F G.M her.F has seen.F/*M

- Non-local dependencies (pronominal binding, syntactic movement)

e.g. cosa_i credi che Maria abbia chiesto a Luigi di comprare _i?

(*what_i do you think (that) M. asked to L. to buy _i?*)

Gianni_i **promette** a Maria_j di _{i/*j} andare a trovarla_{j/k}

Gianni_i **chiede** a Maria_j di _{*i/j} andare a trovarla_{*j/k}

G_i **promises/asks** to M._j _{i/*j *i/j} to go to visit her _{i/k *j/k}

- ⦿ **Adequacy**: a grammar must provide an adequate description of the linguistic reality we want to describe.
- ⦿ We will consider three levels of adequacy:
 - ⦿ **Observational**: the language described by the grammar coincides with the one we want to describe
 - ⦿ **Descriptive**: the grammatical analysis provides relevant structural descriptions that are coherent with the speakers' intuitions
 - ⦿ **Explicative**: the grammar is learnable and it permits to draw conclusions on what's more or less difficult to be processed.

Grammar adequacy

Noam Chomsky



Descriptive levels David Marr



Vision (1981)

- ⦿ **Computational** level: What is the goal of the computation, why is it appropriate, and what is the logic of the strategy by which it can be carried out?
- ⦿ **Representational and algorithmic** level: How can this computational theory be implemented? In particular, what is the representation for the input and output, and what is the algorithm for the transformation?
- ⦿ **Hardware implementational** level: How can the representation and algorithm be realized physically?

Basic formal notions

- ⊙ **Finite sets definition:** $A = \{a, b, c\}$
- ⊙ **Infinite (inductive) set definition:** $A = \{x: x \text{ has a propriety } p\}$
- ⊙ **Ordered sets (n -tuples):** $A = (a, b, c)$
- ⊙ **Cardinality:** $|A| = \text{number of items of } A$

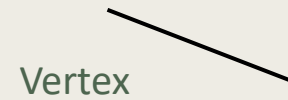
- ⊙ **Cartesian product:** $A = \{a, b, c\}$ $B = \{x, y\}$
 $A \times B = \{(a, x), (b, x), (c, x), (a, y), (b, y), (c, y)\}$
- ⊙ **Union:** $A \cup B = \{x: x \in A \quad \text{or} \quad x \in B\}$
- ⊙ **Concatenation:** $A \circ B = \{xy: x \in A \quad \text{and} \quad y \in B\}$
- ⊙ **Star (Kleene operator):** $A^* = \{x_1 x_2 \dots x_n : n \geq 0 \text{ for any } x_i \in A\}$

Basic formal notions

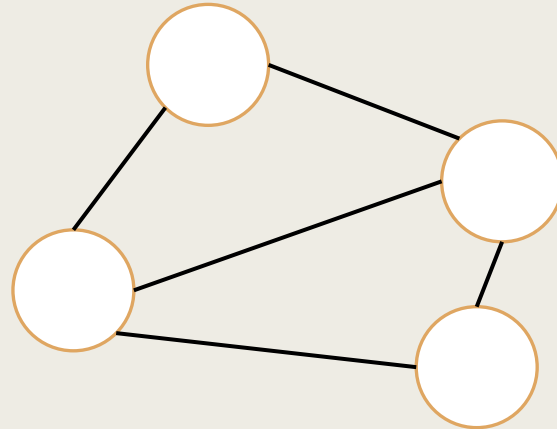
- ⊙ **Indexes:** $x_k = k^{\text{th}}$ element in a series
 $x^k =$ a series of k elements
 $X^R =$ mirror image of X
- ⊙ **Function:** $f(x) \rightarrow y$ ($x =$ Domain, $y =$ Range):
- ⊙ **Predicates:** $f(x) \rightarrow \{\text{true}, \text{false}\}$
- ⊙ **n-places predicates:** $f(x, y \dots z) \rightarrow \{\text{true}, \text{false}\}$
- ⊙ **Equivalence relation:** binary predicates R for which the following properties are valid:
 - ⊙ R is **reflexive**, that is, for any x , xRx ;
 - ⊙ R is **symmetric**, that is, for any x and y , if xRy then yRx ;
 - ⊙ R is **transitive**, that is, for any x , y and z , if xRy and yRz then xRz ;

Basic formal notions

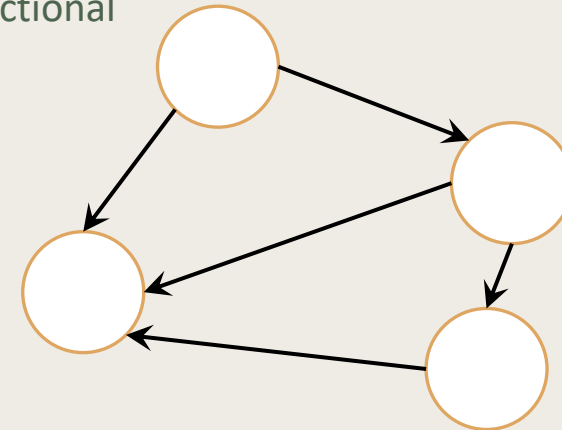
⦿ Graphs



Non directional

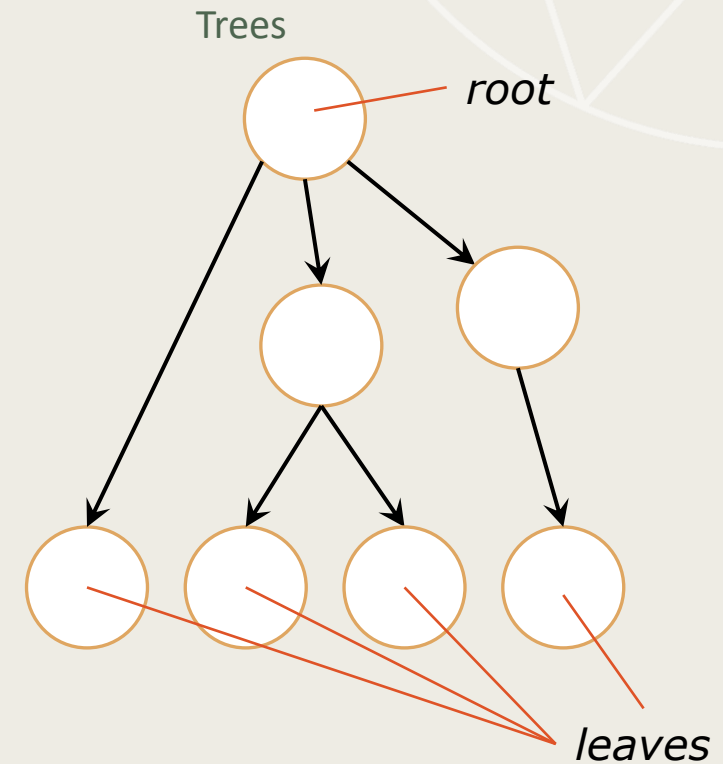
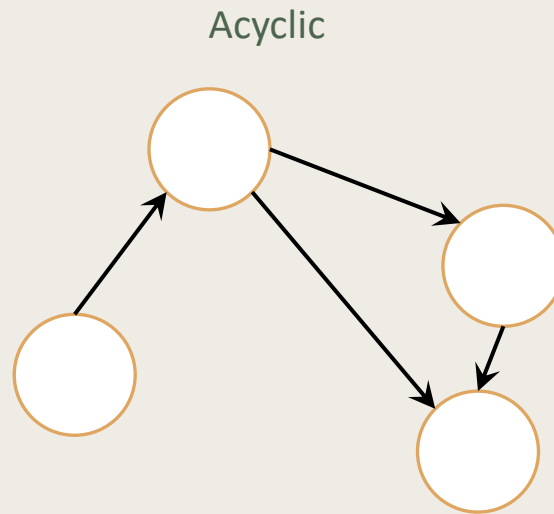
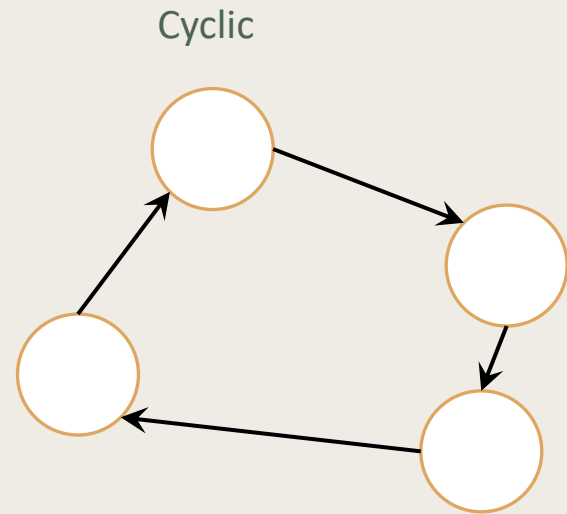


Directional



Basic formal notions

⦿ Graphs



Degree: number of in/out vertices of a node

How to formalize a grammar

- ⊙ **A = Alphabet**

Finite set of chars (A^* = the set of all possible strings built concatenating elements of A ; ε is the null element)

- ⊙ **V = Vocabulary**

(potentially in)finite set of words, built concatenating elements of A
($V \subseteq A^*$)

- ⊙ **L = Language**

(potentially in)finite set of sentences, built concatenating elements of V
($L \subseteq V^*$)

How to formalize a grammar

- ⦿ A **formal grammar** for a language **L** is a set of rules that allows us to **recognize** and **generate** all (and only) the sentences belonging to **L** and (eventually) assign to them an adequate **structural description**.
- ⦿ A Formal Grammar G must be:
 - ⦿ **explicit** (each grammaticality judgment must be just the result of the mechanical application of the rules)
 - ⦿ **consistent** (the very same sentence can't be judged both grammatical and ungrammatical at the same time)

How to formalize a grammar

- ◎ **Phrase Structure Grammar, PSG** (Chomsky 1965)
is an ordered 4-tuple (V_T , V_N , \rightarrow , **Roots**):

V_T is the terminal vocabulary

V_N is the non-terminal vocabulary ($V_T \cup V_N = V$)

\rightarrow is a binary, asymmetric, transitive relation defined on V^* , also known as **rewriting rule**:
for any symbol $A \in V_N$ $\phi A \psi \rightarrow \phi \tau \psi$ for some $\phi, \tau, \psi \in V^*$

Roots is a subset of V_N defined as the axiom(s) of the rewriting rules.
By default, **S (Sentence)** is the only symbol present in this set.

How to formalize a grammar

- ⊙ Give two strings ϕ and $\psi \in V^*$ there is a **ϕ -derivation of ψ** if $\phi \rightarrow^* \psi$.
- ⊙ If there is a ϕ -derivation of ψ then we conclude that **ϕ dominates ψ** . Such a relation is reflexive and transitive.
- ⊙ A ϕ -derivation of ψ is **terminated** if:
 - $\psi \in V_T^*$
 - There is no χ such that a ψ -derivation of χ exists
- ⊙ Given a grammar **G** , a language generated by **G** , is said **$L(G)$** , that is the set ψ of all possible strings for which a terminated S-derivation of ψ exists

Structural description (syntactic tree)

⊙ A **Structural Description** is a 5-tuple

(**V**, **I**, **D**, **P**, **A**) such that:

V is a finite set of vertices (e.g. $v_1, v_2, v_3 \dots$)

I is a finite set of labels (e.g. S, DP, VP, the, table...)

D is a **dominance** relation, which is a **weak relation** (namely a binary, reflexive antisymmetric and transitive relation) defined on **V**

P is a **precedence** relation, which is a **strict order** (namely a binary, anti-reflexive antisymmetric and transitive relation) defined on **V**

A is an **assignment function**;
i.e. a non surjective relation from **V** to **I**

Generative capacity and equivalence

- ⦿ The generative capacity indicates the set of sentences that can be generated; two grammars can be considered **equivalent** in two senses:
 - ⦿ **Weak**, if only the set of sentences is considered
 - ⦿ **Strong**, if we also consider the structural description associated

Decidability

- ⦿ A set Σ is considered
 - ⦿ **decidable** (or recursive) if for any element e , belonging to the **universe set**, there is a **mechanical procedure** that **in a finite set of steps terminates** by saying if $e \in$ or \notin to Σ (not belonging to Σ implies that e belongs to the complement of Σ defined as $\bar{\Sigma}$)
 - ⦿ **Recursively enumerable** when a procedure exists that enumerates all and only the elements of Σ

Regular Grammars / Languages

⦿ **Regular grammars** admit rules of this kind:

$A \rightarrow xB$

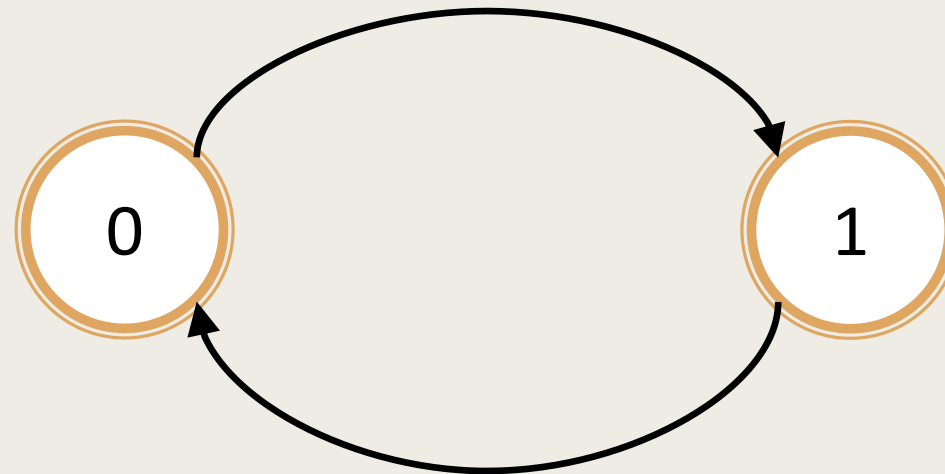
Or (systematically) of this kind:

$A \rightarrow Bx$

The languages generated by such grammars are named **Regular**

Automata and computation

- ⦿ **Automata** are mathematical computational models composed by states and transitions among states
- ⦿ Example of automaton: electric switch!
 - 0 = on
 - 1 = off
 - -> = push



Finite State Automata (FSA)

⊙ Finite-State Automata (FSA)

are 5-tuples $\langle Q, \Sigma, q_0, F, \delta \rangle$ where:

Q = non-null, finite set of states

Σ = non-null, finite set of characters (alphabet) acceptable as input

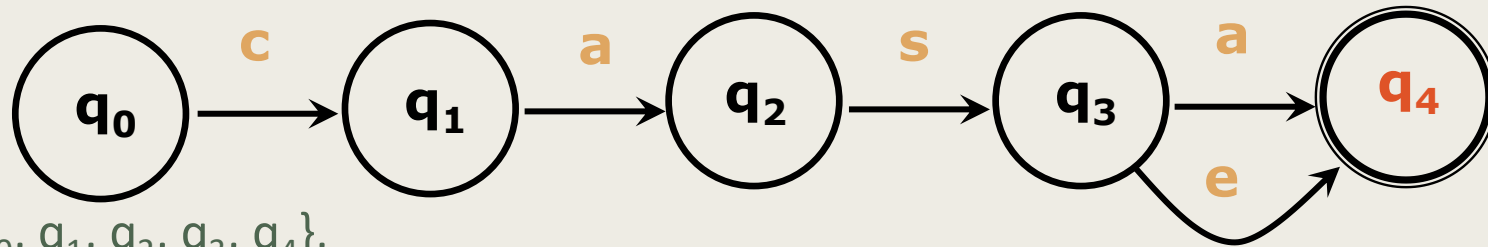
q_0 = initial states, such that $q_0 \in Q$

F = final states, such that $F \in Q$

δ = finite set of rules defined from $Q \times \Sigma$ to Q

FSA as word processors

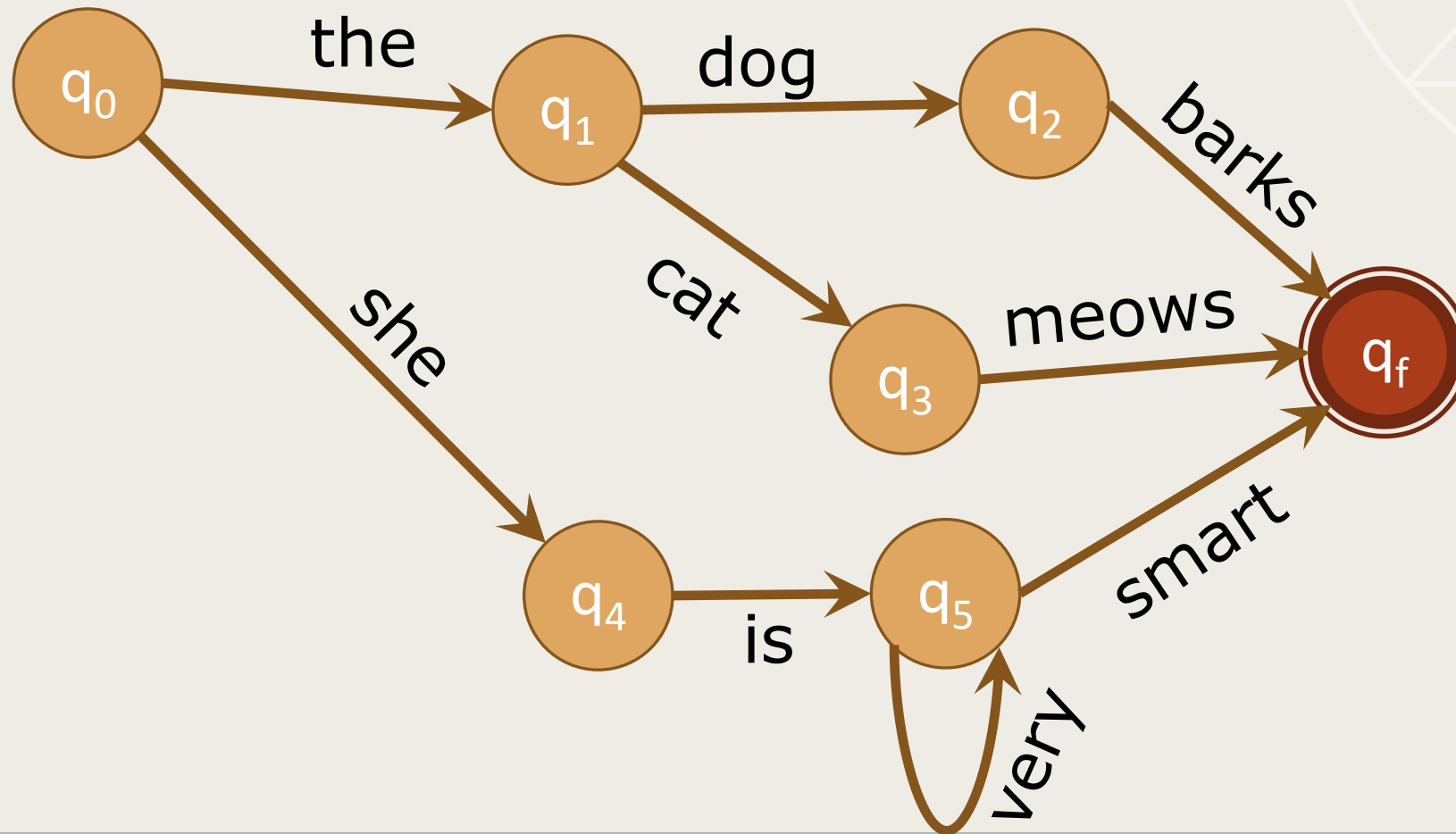
- ⦿ A set of FSA is not just a mechanism that recognizes or generates a lexical item, but it might represent a sentence (or a language...).
- ⦿ FSA recognizing word casa and its plural form:



- ⦿ $Q = \{q_0, q_1, q_2, q_3, q_4\}$,
- ⦿ $\Sigma = \{c, a, s, e, \#\}$,
- ⦿ $Q_0 = \{q_0\}$,
- ⦿ $F = \{q_4\}$,
- ⦿ $\delta =$

	q_0	q_1	q_2	q_3	q_4
c	q_1				
a		q_2		q_4	
s			q_3		
e				q_4	

Finite State Grammars (Regular Languages)



Regular Expressions (RE)

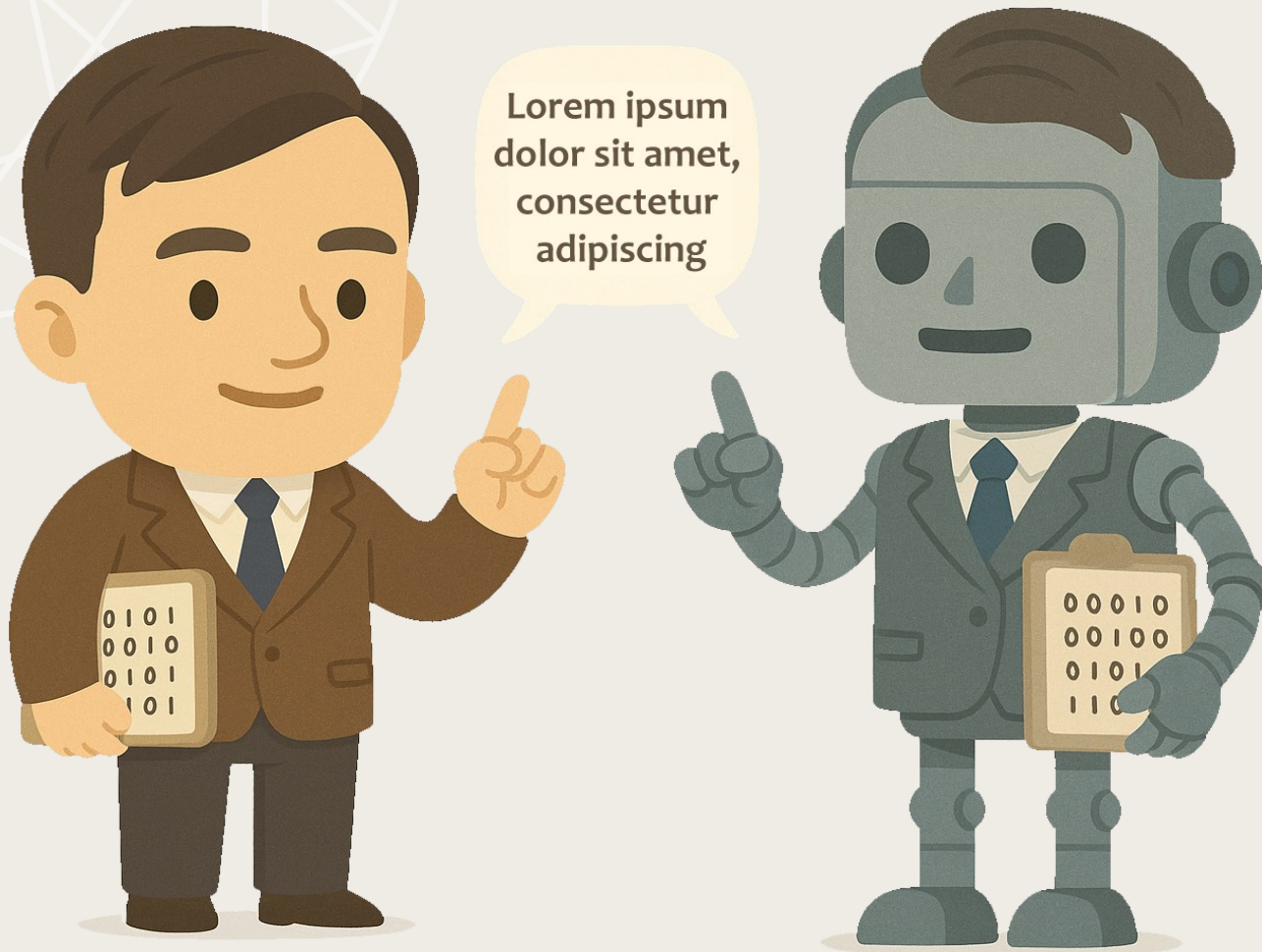
- ⦿ We can use **Regular Expressions** to query corpora for instance.
RE are algebraic notation to express sets of strings.
- ⦿ The key of a **RE** is the **identification pattern** composed by alphanumeric chars (including spacing and punctuation) and special signs indicating precise relations among chars.

RE	Correspondence	E.g. Identified pattern
[Cc]athedral	<u>Cathedral</u> or <u>cathedral</u>	The <u>cathedral</u> is gorgeous
[^a-z]	All but not low chars	<u>I</u> l duomo è ...
sai?d	<u>said</u> or <u>sad</u>	You must be <u>sad</u>
ma.t	Any char between <i>a</i> and <i>t</i>	<u>malt</u> or <u>matt</u>
bu*	<i>b</i> followed by any (even null) number of <i>u</i>	buuuuuu! or b!
^T Vs. e\$	^ = string beginning; \$ = string end	<u>The</u> house
passe(s d)	expresses the logical disjunction	<u>passes</u> or <u>passed</u>
*	Escape char	A <u> </u> *

RG, FSA and RE equivalence

- ⦿ **Regular Grammars (RG), Finite State Automata (FSA) and Regular Expressions (RE) are equivalent, i.e. they describe the very same set of languages: **Regular Languages**.**
- ⦿ Proof by construction

Computing Machinery and Intelligence (1950)



The imitation game Alan Turing



Do you know Eliza?



● Eliza (Weizenbaum, anno 1966):

- Human: Well, my boyfriend made me come here.
- **ELIZA:** *Your boyfriend made you come here?*
- Human: He says I'm depressed much of the time.
- **ELIZA:** *I am sorry to hear you are depressed.*
- Human: It's true. I'm unhappy.
- **ELIZA:** *Do you think coming here will help you not to be unhappy?*

Eliza uses regular expressions!

⦿ RE and Substitution

- s/Regular_Expression_1/Regular_Expression_2/
- s/ [a-z]* / blah /

⦿ Registers: using block operators (round brackets indicates a block), we can reuse a matched pattern:

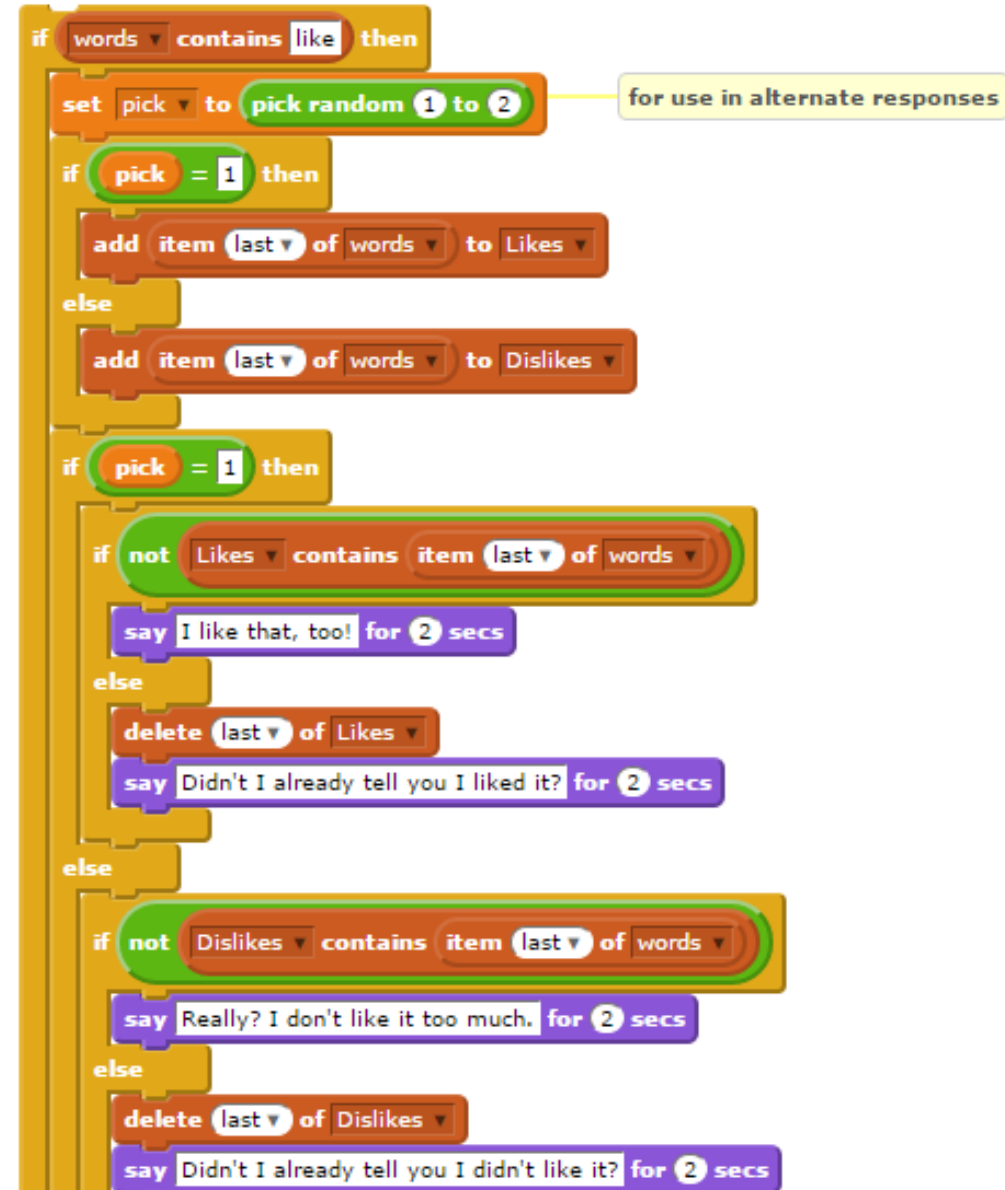
- s/ the (house|car) has been bought by (Mary|John)/ \2 bought the \1 /

⦿ Substitutions by ELIZA:

- s/ I'm [.* ?](depressed|sad)/I'm sorry to hear that you are \1/
- s/ everybody is (.*) / in which sense they are \1?/
- s/ always / can you make a specific example?

Trivia: chatbots

Programming a chatbot
for *like/dislike* patterns
using «Scratch»



Trivia: ChatGPT (GPT-3.5)

<https://openai.com/api/>

PE

Write a limerick wishing Noam Chomsky a happy 94th birthday.



There once was a man named Noam
Whose mind was as sharp as a poem
He's turning ninety-four
And we couldn't be more
Excited to celebrate his happy birthday, oh Noam!

With his theories on language and thought
He's taught us a thing or two that we ought
To know about the mind
And the ways it's designed
So here's to Noam, our brilliant and wise old tot!

May your day be filled with joy and cheer
And may the coming year be filled with good cheer
Happy birthday, Noam, you truly are
A treasure to the world of linguistics, by far!

PE

Have more people been to Berlin than I have?



It is impossible for me to say for sure whether more people have been to Berlin than you have. This is because I am a large language model trained by OpenAI and do not have the ability to access information about your personal experiences or the experiences of others. In general, however, Berlin is a popular tourist destination and it is likely that many people have visited the city.



Is ChatGPT “different” from Eliza?

On the Dangers Can Language M

Emily M. Bender*

Last 6 months

1. Piantadosi - Modern language models refute Chomsky’s approach to language
2. Agodio et al. - Indexed definiteness without demonstratives in Guébie
3. Müller - Large Language Models: The best linguistic theory, a wrong linguistic theory, or no linguistic theory at all?
4. Manova - Machine learning versus human learning: Basic units and form-meaning mapping
5. Ches
6. Haid
7. Dou
8. Man
9. Gree
10. Ch

Modern language models refute Chomsky’s approach to language

Steven Piantadosi

July 2024

Modern machine learning has subverted and bypassed the theoretical framework of Chomsky’s generative approach to linguistics, including its core claims to particular insights, principles, structures, and processes. I describe the sense in which modern language models implement genuine theories of language, and I highlight the links between these models and approaches to linguistics that are based on gradient computations and memorized constructions. I also describe why these models undermine strong claims for the innateness of language and respond to several critiques of large language models, including arguments that they can’t answer “why” questions and skepticism that they are informative about real life acquisition. Most notably, large language models have attained remarkable success at discovering grammar without using any of the methods that some in linguistics insisted were necessary for a science of language to progress. (UPDATED: With a postscript on replies to the original draft)

Format: [pdf]

Reference: lingbuzz/007180
(please use that when you cite this article)

Published in: Edward Gibson & Moshe Poliak (eds.), From fieldwork to linguistic theory: A tribute to Dan Everett (Empirically Oriented Theoretical Morphology and Syntax 15), 353–414. Berlin: Language Science Press. <https://doi.org/10.5281/zenodo.12665933>.

keywords: large language model, minimalism, chomsky, generative syntax, emergent, computational modeling, statistical learning, cognitive science, syntax

previous versions:
v7 [November 2023]
v6 [October 2023]
v5 [September 2023]
v4 [March 2023]
v3 [March 2023]
v2 [March 2023]
v1 [March 2023]

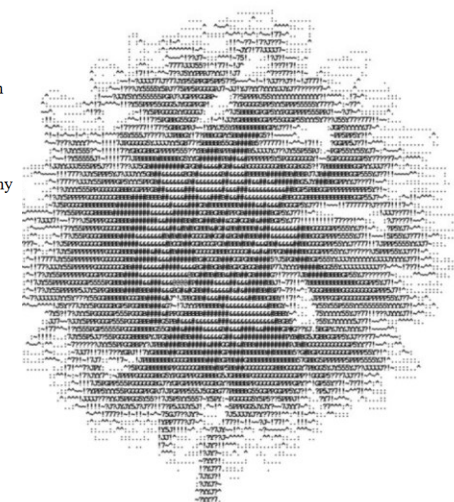
Downloaded: 31240 times

The New York Times

OPINION
GUEST ESSAY

Noam Chomsky: The False Promise of ChatGPT

March 8, 2023



msky, Ian Roberts and Jeffrey Watumull

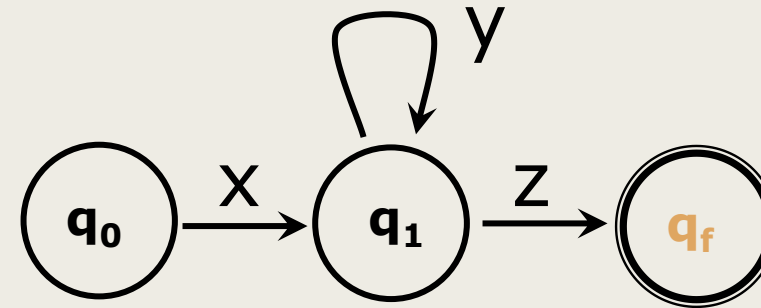
Dr. Chomsky and Dr. Roberts are professors of linguistics. Dr. Watumull is a director of artificial intelligence at a science and technology company.

How to determine if a string can be generated by a Regular Grammar?

⊙ Pumping lemma for Regular Grammar

If A is a Regular Language, then there is a number p (expressing «pumping» magnitude), for which, if s is a generic string A of length at least equal to p , then it can be split in 3 parts, $s = xyz$ such that:

- I. For any $i \geq 0$, $xy^iz \in A$
- II. $|y| > 0$
- III. $|xy| \leq p$



- ⊙ $a^n b^n$ (counting recursion) cannot be generated by Regular Grammars (no way to pump a number of a s followed by the very same number of b s)

Context-Free Grammars

- ⦿ **Context-Free Grammars (CFG)** admits only this kind of rules:

$A \rightarrow \gamma$ (where γ is any sequence of (non)terminal symbols)

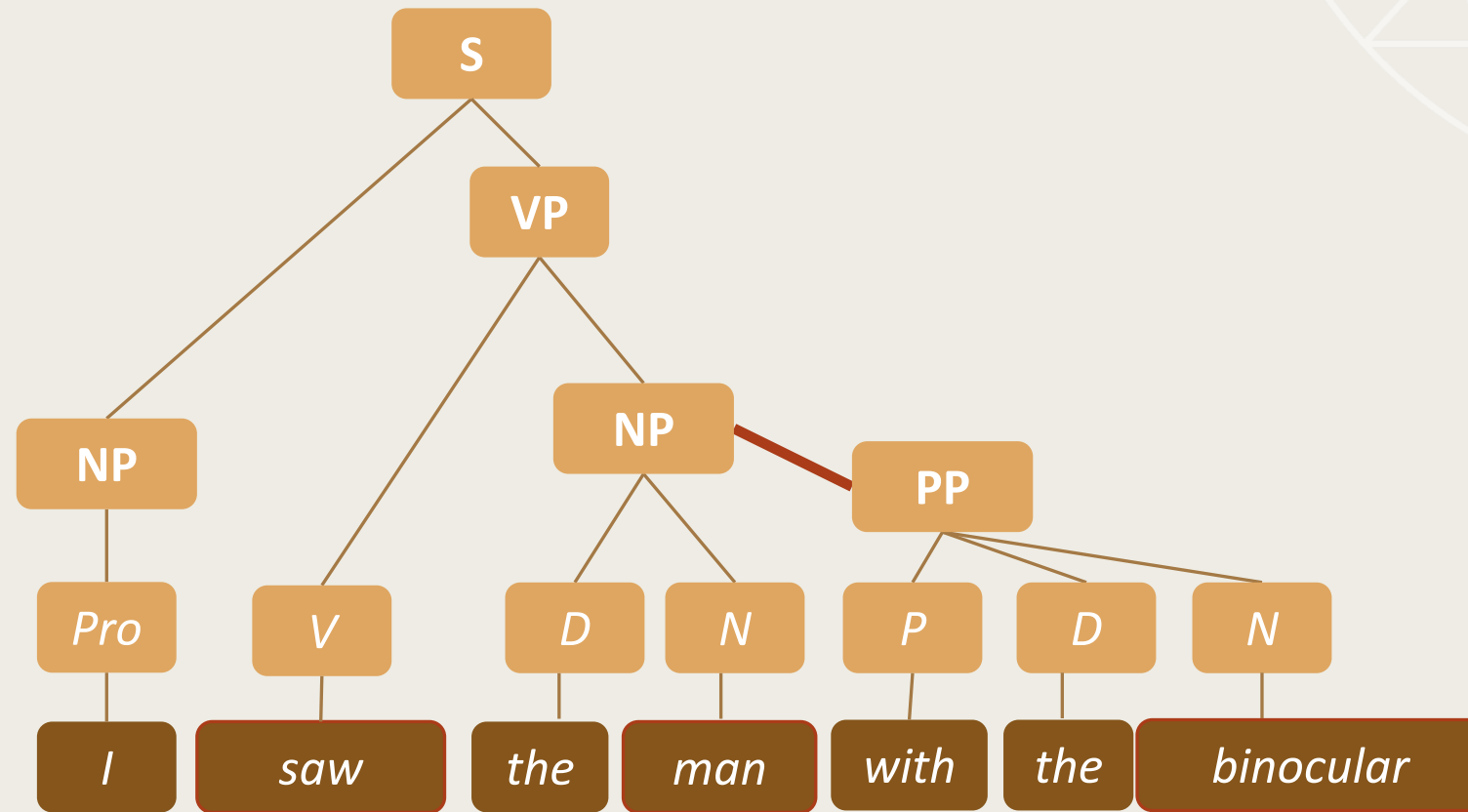
Languages generated by CFG are named **Context-Free Languages**

- ⦿ Any **CFG** can be «converted» in a (weakly) equivalent CFG in the **Chomsky Normal Form (CNF)**:

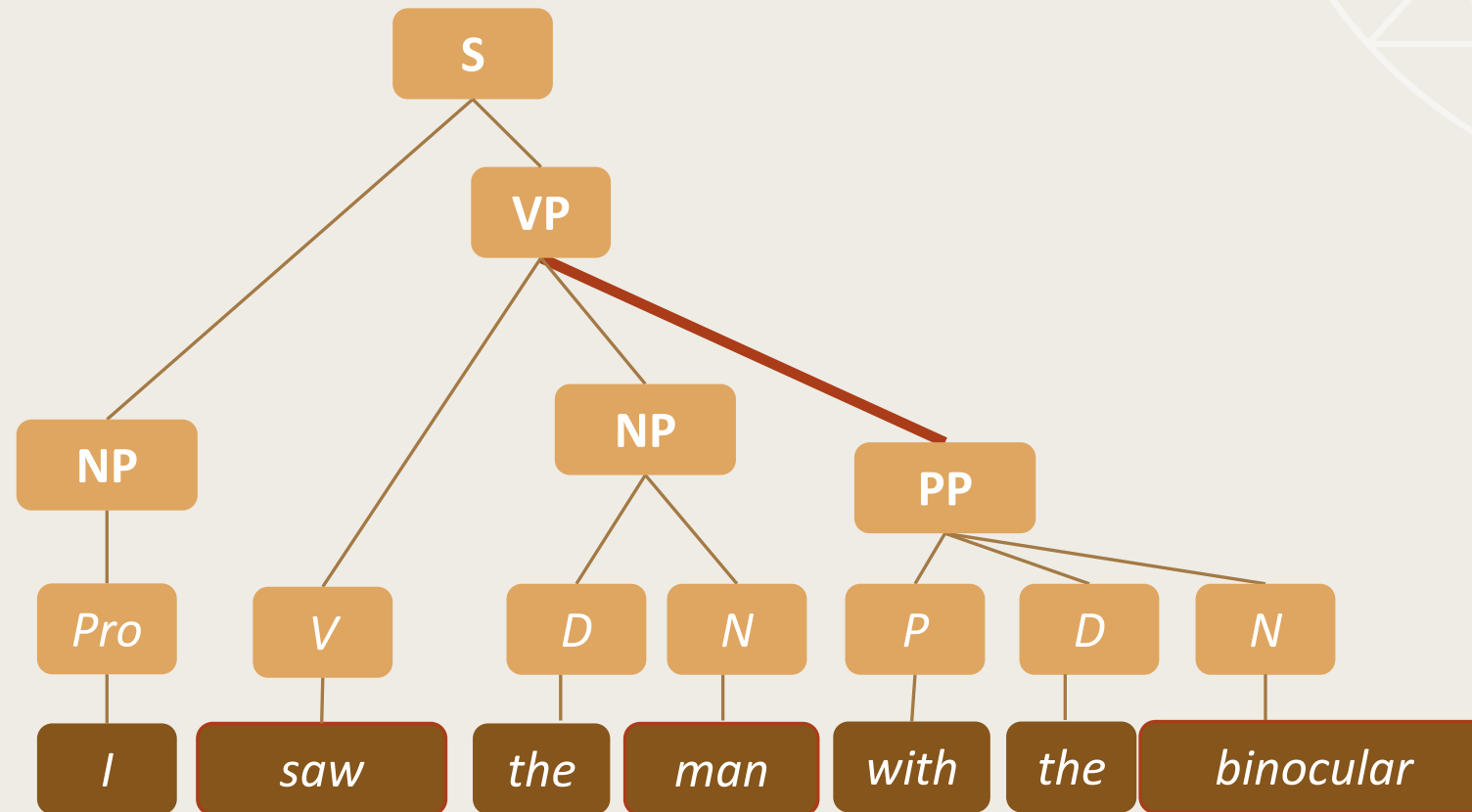
$A \rightarrow BC$

$A \rightarrow a$

Describing syntactic ambiguity



Describing syntactic ambiguity



Describing syntactic ambiguity

- ⦿ Rules with the same left-side symbol should be present in the grammar to permit ambiguity:

- ⦿ $VP \rightarrow V\ NP$
- ⦿ $VP \rightarrow V\ NP\ PP$

- ⦿ $NP \rightarrow D\ N$
- ⦿ $NP \rightarrow D\ N\ PP$

Push-Down Automata

- ⊙ A **Push-Down Automata** (PDA) is a finite state automata endowed with a memory **stack**; PDAs are defined by 6-tuples $\langle Q, \Sigma, q_0, F, \delta, \Gamma \rangle$ where:

Q = finite and non-null set of states

Σ = finite and non-null set of characters accepted as input (alphabet)

q_0 = initial state(s), such that $q_0 \in Q$

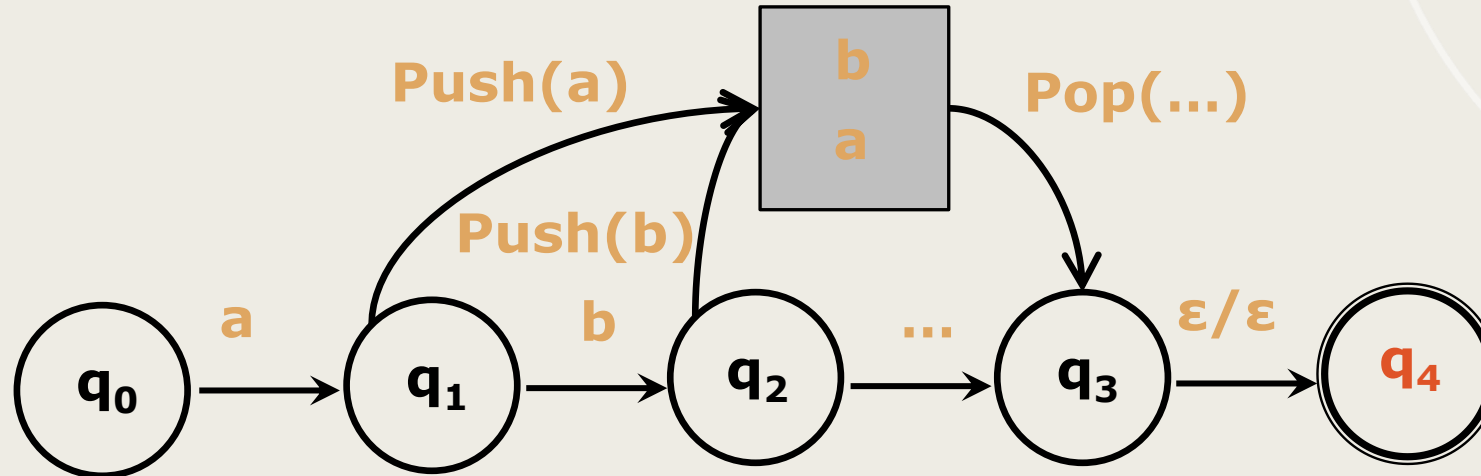
F = final states(s), such that $F \in Q$

δ = finite and non-null set of transitional rules defined from $Q \times \Sigma \times \Gamma$ to $Q \times \Gamma$

Γ = finite and non-null set of characters that can be stored in memory (Γ can have the same symbols as Σ)

PDA can parse mirror recursion

⊙ XX^R



- ⊙ $Q = \{q_0, q_1, q_2, q_3, q_4\},$
- ⊙ $\Sigma / \Gamma = \{a, b, \epsilon\},$
- ⊙ $Q_0 = \{q_0\},$
- ⊙ $F = \{q_4\},$
- ⊙ $\delta =$

	q_0	q_1	q_2	q_n	q_4
a	q_1 <i>push(a)</i>				
b		q_2 <i>push(b)</i>			
\dots			q_n <i>pop(...)</i>		
ϵ				q_4	

CFG and PDA equivalence

- ◎ **Context-Free Grammars (CFG)**, and **Push-Down Automata (PDA)** are equivalent (i.e. they describe the very same set of languages: the Context-Free Languages).

«Demonstration» by construction:

1. For any S rule, create a PDA q_0 rule such that:
 $(q_0, \epsilon, \epsilon) \rightarrow (q_1, S)$
2. For any other CFG rule such that $A \rightarrow x$, create PDA rules such that:
 $(q_1, \epsilon, A) \rightarrow (q_1, x)$
3. For any symbol $a : a \in V_T$, create PDA rules such that:
 $(q_1, a, a) \rightarrow (q_1, \epsilon)$

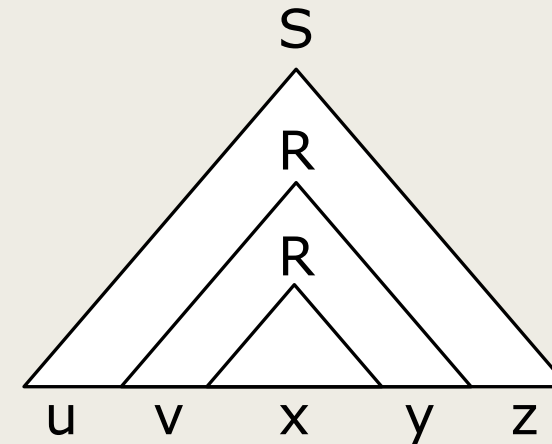
Limits of CFGs?

⊙ Pumping lemma for Context-Free Grammars

If A is a Context-Free Language, then there is a number p (expressing the «pumping» length), for which, if s is a string of A of length at least equal to p , then it can be divided in 5 parts,

$s = uvxyz$ such that:

- I. For any $i \geq 0$, $uv^i xy^i z \in A$
- II. $|vy| > 0$
- III. $|vxy| \leq p$



⊙ E.g. neither $a^n b^n c^n$ nor XX is not generable by CFGs.

Inclusion relations among Grammars

⊙ Chomsky's Hierarchy (1956, 59):

Type 3: Regular Grammars (equivalent device: Finite State Automata)

$A \rightarrow xB$

Type 2: Context Free Grammars (equivalent device: Push-Down Automata)

$A \rightarrow \gamma$

Type 1: Context Sensitive Grammars (e.g.: Linear-Bounded Automata)

$\alpha A \beta \rightarrow \alpha \gamma \beta \quad (\gamma \neq \varepsilon)$

Type 0: Turing Equivalent Grammars (e.g. Augmented Transition Networks)

$\alpha \rightarrow \beta \quad (\alpha \neq \varepsilon)$

La Gerarchia di Chomsky



THE ALGEBRAIC THEORY OF CONTEXT-FREE LANGUAGES*

N. CHOMSKY

Massachusetts Institute of Technology

AND

M. P. SCHÜTZENBERGER

Harvard University

* This work was supported in part by the U. S. Army Signal Corps, the Air Force Office of Scientific Research, and the Office of Naval Research; and in part by the National Science Foundation; and in part by a grant from the Commonwealth Fund.

Recursive enumerable (Turing Machines)

Context-sensitive

Context-free

Regular



ab^*



$a^n b^n$



$a^n b^n c^n$

Where are Natural Languages?

- ⊙ Natural languages are **NOT generable** by **Regular Grammars** (Chomsky 1956):

If X then Y (with A and B potentially of the form “if X then Y”, generating then a **counting dependency** of the $a^n b^n$ kind, that is: **$if^n then^n$**)

- ⊙ Natural languages are **NOT even generable** by **Context-Free Grammars** (Shieber 1985):

Jan säit das mer em Hans es huus hälfed aastriiche
 (“famous” Swiss-German dialect)
J. says that we to H. The house have helped painting

Gianni, Luisa e Mario sono rispettivamente
sposato, divorziata e scapolo
(“**ABC...ABC**”... Are languages of the **XX** kind)

Where are Natural Languages?

◎ **Recursion** in natural languages (that is, how to make infinite use of finite means):

● **Right recursion** (ab^n : iteration or «tail recursion»):
[the dog bit [the cat [that chased [the mouse [that ran]]]]]

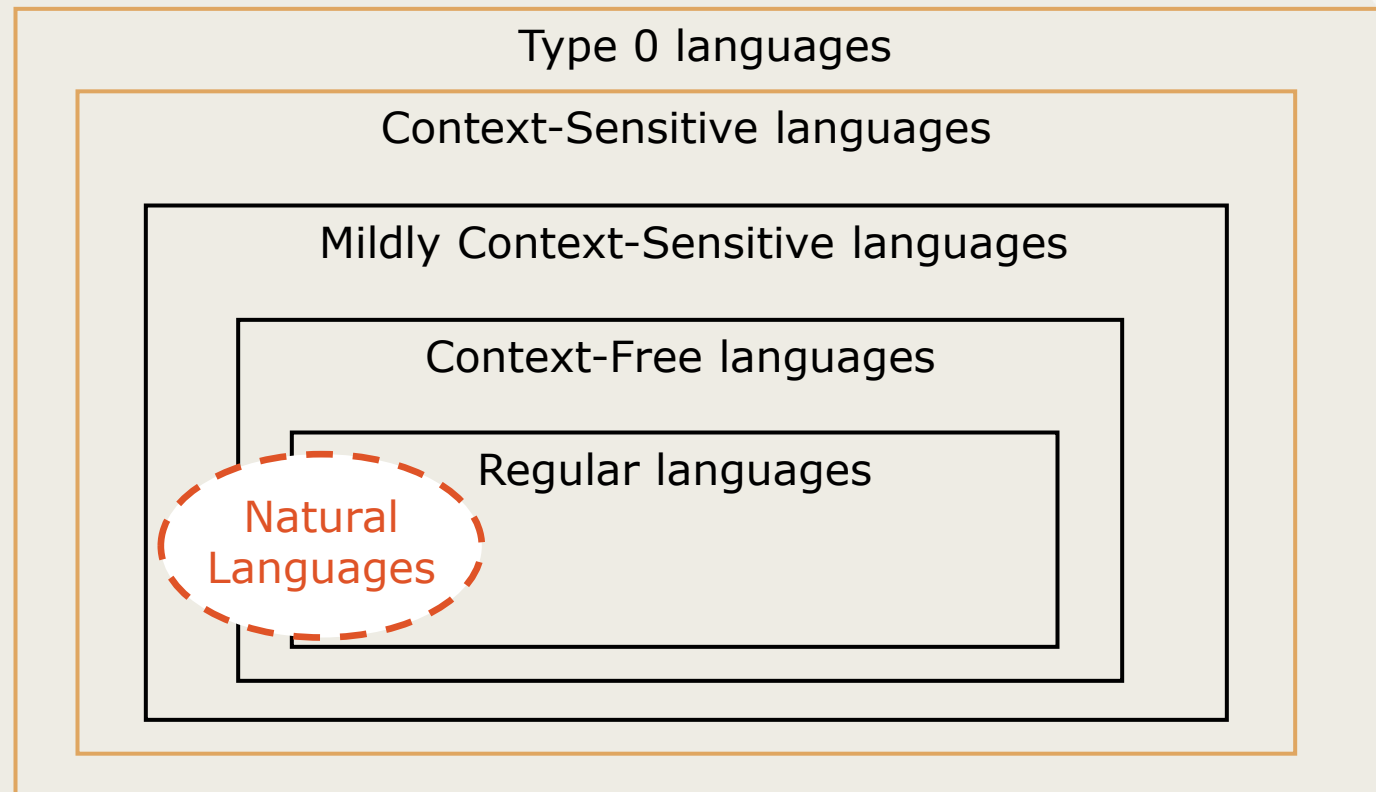
● **Center embedding** ($a^n b^n$: counting recursion or «true recursion»):
[the mouse [(that) the cat [(that) the dog bit] chased] ran]

● **Cross-serial dependencies** (xx, identity recursion)

Aldo, Bea e Carlo sono rispettivamente sposato, nubile e divorziato

$A_{.male}$, $B_{.female}$, $C_{.male}$ are respectively married_{male}, unmarried_{female} & divorced_{male}

Where are Natural Languages?



Today's key concepts

⦿ What's a **formal grammar**

- ⦿ **Rewriting Rules** and **Recursion**
- ⦿ Rewriting Rules **restrictions** create grammar classes organized in an inclusion hierarchy (**Chomsky's Hierarchy**)
- ⦿ **Regular Grammars** (RG), **Regular Expressions** (RE) and **Finite State Automata** (FSA) **equivalence**
- ⦿ **Context-Free Grammars** (CFG) and **Push-Down Automata** (PDA) **equivalence**
- ⦿ Using **pumping lemmas** to decide if a certain string property can be captured or not by a certain class of grammars
- ⦿ **Natural languages** are **neither** Regular, **nor** Context-Free (though RGs and CFGs are often used to process Natural Languages!)