# Formal Languages and Linguistics

Pascal Amsili

Sorbonne Nouvelle, Lattice (CNRS/ENS-PSL)

Cogmaster, september 2020

# Overview

1. Formal Languages
2. Formal Grammars
3. Regular Languages
   - Definition
   - Automata
   - Properties
4. Formal complexity of Natural Languages

# Definition

3 possible definitions

1. a regular language can be generated by a regular grammar
2. a regular language can be defined by rational expressions
3. a regular language can be recognized by a finite automaton

**Def. 15 (Rational Language)**

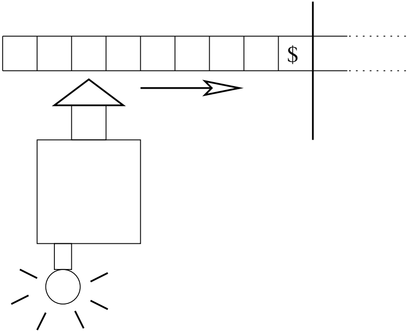
A rational language on $\Sigma$ is a subset of $\Sigma^*$ inductively defined thus:

- $\emptyset$ and $\{\varepsilon\}$ are rational languages ;
- for all $a \in X$, the singleton $\{a\}$ is a rational language ;
- for all $g$ and $h$ rational, the sets $g \cup h$, $g.h$ and $g^*$ are rational languages.

# Overview

1. Formal Languages
2. Formal Grammars
3. Regular Languages
   - Definition
   - Automata
   - Properties
4. Formal complexity of Natural Languages

# Metaphoric definition



# Formal definition

## Def. 16 (Finite deterministic automaton (FDA))

A finite state deterministic automaton $\mathcal{A}$ is defined by :

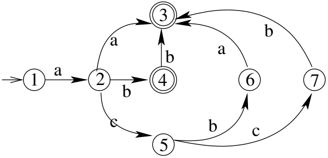
$$\mathcal{A} = \langle Q, \Sigma, q_0, F, \delta \rangle$$

- $Q$ is a finite set of states
- $\Sigma$ is an alphabet
- $q_0$ is a distinguished state, the initial state,
- $F$ is a subset of $Q$, whose members are called final/terminal states
- $\delta$ is a mapping **fonction** from $Q \times \Sigma$ to $Q$. Notation $\delta(q, a) = r$.

# Example

Let us consider the (finite) language $\{aa, ab, abb, acba, accb\}$.
The following automaton recognizes this langage: $\langle Q, \Sigma, q_0, F, \delta \rangle$, avec $Q = \{1, 2, 3, 4, 5, 6, 7\}$, $\Sigma = \{a, b, c\}$, $q_0 = 1$, $F = \{3, 4\}$, and $\delta$ is thus defined:

$\delta$ :
(1,a) $\mapsto$ 2
(2,a) $\mapsto$ 3
(2,b) $\mapsto$ 4
(2,c) $\mapsto$ 5
(4,b) $\mapsto$ 3
(5,b) $\mapsto$ 6
(5,c) $\mapsto$ 7
(6,a) $\mapsto$ 3
(7,b) $\mapsto$ 3




| | a | b | c |
|---|---|---|---|
| $\rightarrow$ 1 | 2 | | |
| 2 | 3 | 4 | 5 |
| $\leftarrow$ 3 | | | |
| $\leftarrow$ 4 | | 3 | |
| 5 | | 6 | 7 |
| 6 | 3 | | |
| 7 | | 3 | |

# Recognition

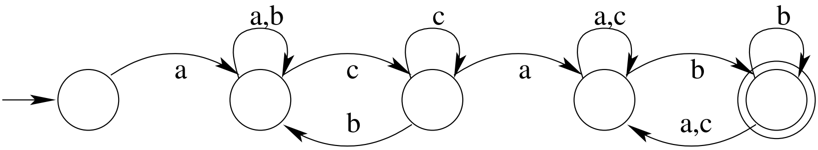
Recognition is defined as the existence of a sequence of states defined in the following way. Such a sequence is called a path in the automaton.

## Def. 17 (Recognition)

A word $a_1 a_2 ... a_n$ is **recognized/accepted** by an automaton iff there exists a sequence $k_0, k_1, ..., k_n$ of states such that:

$$k_0 = q_0$$
$$k_n \in F$$
$$\forall i \in [1, n], \quad \delta(k_{i-1}, a_i) = k_i$$

# Example



# Overview

1. Formal Languages
2. Formal Grammars
3. Regular Languages
   - Definition
   - Automata
   - Properties
4. Formal complexity of Natural Languages

# Pumping lemma: Intuition

Take an automaton with $k$ states.

# Pumping lemma: Intuition

Take an automaton with $k$ states.
If the accepted language is infinite,
then some words have more than $k$ letters.

# Pumping lemma: Intuition

Take an automaton with $k$ states.
If the accepted language is infinite,
then some words have more than $k$ letters.
Therefore, at least one state has to be “gone through” several times.

# Pumping lemma: Intuition

Take an automaton with $k$ states.
If the accepted language is infinite,
then some words have more than $k$ letters.
Therefore, at least one state has to be "gone through" several times.
That means there is a loop on that state.

# Pumping lemma: Intuition

Take an automaton with $k$ states.
If the accepted language is infinite,
then some words have more than $k$ letters.
Therefore, at least one state has to be "gone through" several times.
That means there is a loop on that state.
Then making any number of loops will end up with a word in L.

$\Rightarrow$ Pumping lemma

# Pumping lemma: definition

**Def. 18 (Pumping Lemma)**

Let $L$ be an infinite regular language.
There exists an integer $k$ such that:

$$\forall x \in L,\ |x| > k,\ \exists u, v, w \text{ such that } x = uvw,\ \text{with:}$$
$$(i)\quad |v| \geq 1$$
$$(ii)\quad |uv| \leq k$$
$$(iii)\quad \forall i \geq 0,\ uv^iw \in L$$

# Pumping lemma: Illustration

Let's illustrate the lemma with a language which trivialy satisfies it: $a^*bc$.

Let $k = 3$, the work $abc$ is long enough, and can be decomposed:

$\underbrace{\varepsilon}_{u} \quad \underbrace{a}_{v} \quad \underbrace{b \quad c}_{w}$

The three properties of the lemma are satisfied:

- $|v| \geq 1$ $(v = a)$
- $|uv| \leq k$ $(uv = a)$
- $\forall i \in \mathbb{N}$, $uv^iw(= a^ibc)$ belongs to the language by definition.

# Pumping lemma: Consequences

The pumping lemma is a tool to prove that a language is **not** regular.

| | | |
|---|---|---|
| $\mathcal{L}$ regular | $\Rightarrow$ | pumping lemma $(\forall i, uv^i w \in \mathcal{L})$ |
| pumping lemma | $\not\Rightarrow$ | $\mathcal{L}$ regular |

# Pumping lemma: Consequences

The pumping lemma is a tool to prove that a language is **not** regular.

| | | |
|---|---|---|
| $\mathcal{L}$ regular | $\Rightarrow$ | pumping lemma ($\forall i, uv^iw \in \mathcal{L}$) |
| pumping lemma | $\not\Rightarrow$ | $\mathcal{L}$ regular |

to prove that $\mathcal{L}$ is

- **regular** provide an automaton
- **not regular** show that the pumping lemma does not apply

# Pumping lemma: Consequences

## Def. 19 (Consequences)

Let $\mathcal{A}$ be a $k$ state automaton:

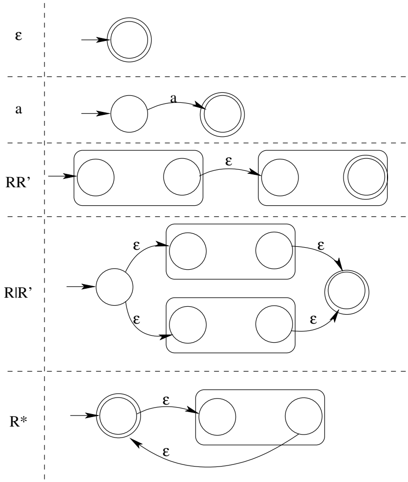
1. $L(\mathcal{A}) \neq \emptyset$ ***iff*** $\mathcal{A}$ recognises (at least) one word $u$ s.t. $|u| < k$.
2. $L(\mathcal{A})$ is infinite ***iff*** $\mathcal{A}$ recognises (at least) one word $u$ t.q. $k \leq |u| < 2k$.

# Closure

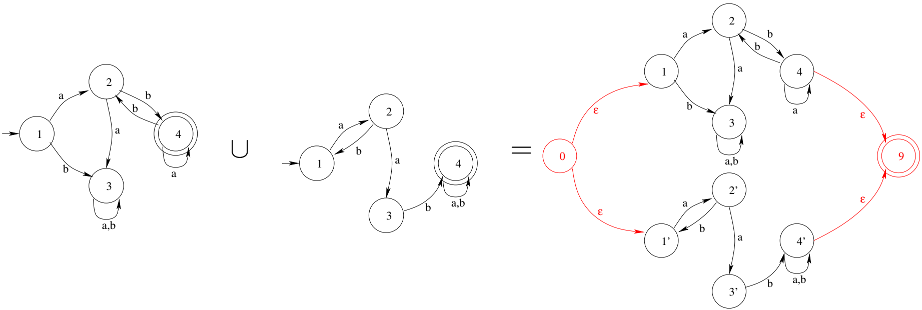
Regular languages are closed under various operations: if the languages $L$ and $L'$ are regular, so are:

- $L \cup L'$ (union); $L.L'$ (product); $L^*$ (Kleene star)
  *(rational operations)*
- $L \cap L'$ (intersection); $\overline{L}$ (complement)
- ...

# Rational operations



# Union of regular languages: an example



# Intersection of regular languages

Algorithmic proof
Deterministic complete automata

| $L_1$ | a | b |
|---|---|---|
| $\rightarrow$ 1 | 2 | 4 |
| 2 | 4 | 3 |
| $\leftarrow$ 3 | 3 | 3 |
| 4 | 4 | 4 |

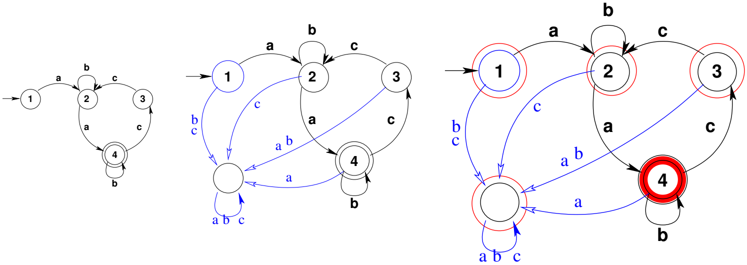
| $L_2$ | a | b |
|---|---|---|
| $\leftrightarrow$ 1 | 2 | 5 |
| 2 | 5 | 3 |
| 3 | 4 | 5 |
| 4 | 1 | 4 |
| 5 | 5 | 5 |

| $L_1 \cap L_2$ | a | b |
|---|---|---|
| $\rightarrow$ (1,1) | (2,2) | (4,5) |
| (2,2) | (4,5) | (3,3) |
| (4,5) | (4,5) | (4,5) |
| (3,3) | (3,4) | (3,5) |
| (3,4) | (3,1) | (3,4) |
| $\leftarrow$ (3,1) | (3,2) | (3,4) |
| (3,2) | (3,4) | (3,3) |
| (3,5) | (3,5) | (3,5) |

# Complement of a regular language

Deterministic complete automata

completed complemented



# Results: expressivity

- Any finite langage is regular
- $a^n b^m$ is regular
- $a^n b^n$ is not regular
- $ww^R$ is not regular ($^R$ : reverse word)

# Decidable problems

- The "word problem" $w \overset{?}{\in} L(\mathcal{A})$ is decidable.

$\Rightarrow$ A computation on an automaton always stops.

# Decidable problems

- The "word problem" $w \overset{?}{\in} L(\mathcal{A})$ is decidable.

$\Rightarrow$ A computation on an automaton always stops.

- The "emptiness problem" $L(\mathcal{A}) \overset{?}{=} \emptyset$ is decidable.

$\Rightarrow$ It's enough to test all possible words of length $\leq k$, where $k$ is the number of states.

# Decidable problems

- The "word problem" $w \stackrel{?}{\in} L(\mathcal{A})$ is decidable.

$\Rightarrow$ A computation on an automaton always stops.

- The "emptiness problem" $L(\mathcal{A}) \stackrel{?}{=} \emptyset$ is decidable.

$\Rightarrow$ It's enough to test all possible words of length $\leq k$, where $k$ is the number of states.

- The "finiteness problem" $L(\mathcal{A}) \stackrel{?}{\text{is}}$ *finite* is decidable.

$\Rightarrow$ Test all possible words whose length is between $k$ and $2k$. If there exists $u$ s.t. $k < |u| < 2k$ and $u \in L(\mathcal{A})$, then $L(\mathcal{A})$ is infinite.

# Decidable problems

- The "word problem" $w \overset{?}{\in} L(\mathcal{A})$ is decidable.

$\Rightarrow$ A computation on an automaton always stops.

- The "emptiness problem" $L(\mathcal{A}) \overset{?}{=} \emptyset$ is decidable.

$\Rightarrow$ It's enough to test all possible words of length $\leq k$, where $k$ is the number of states.

- The "finiteness problem" $L(\mathcal{A}) \overset{?}{\text{is}}$ *finite* is decidable.

$\Rightarrow$ Test all possible words whose length is between $k$ and $2k$. If there exists $u$ s.t. $k < |u| < 2k$ and $u \in L(\mathcal{A})$, then $L(\mathcal{A})$ is infinite.

- The "equivalence problem" $L(\mathcal{A}) \overset{?}{=} L(\mathcal{A}')$ is decidable.

$\Rightarrow$ it boils down to answering the question:
$\left(L(\mathcal{A}) \cap \overline{L(\mathcal{A}')}\right) \cup \left(L(\mathcal{A}') \cap \overline{L(\mathcal{A})}\right) = \emptyset$

# Overview

1. Formal Languages
2. Formal Grammars
3. Regular Languages
4. Formal complexity of Natural Languages
   - Introduction
   - Are NL regular?
   - Are NL context-free?
   - Are NL context-sensitive?

# Motivation

Why an inquiry into the formal complexity of Natural Language(s) ?

- It gives us knowledge about the **structure** of natural languages,
- It helps us assess the **adequation** of linguistic formalisms,
- It gives bound for the **complexity** of NLP tasks,
- It provides us with **predictions** about human language processing.

# Hypotheses

We assume that:

- We can talk about "natural language" in general: all languages have a similar structure, a similar power
- Natural languages are recursively enumerable, i.e. they are formal languages
- Natural languages are infinite

$\Rightarrow$ Under these hypotheses, it is possible to ask the question: what is the complexity of natural languages ?

# An infinite number of sentences

1. Arbitrary long sentences can be built by adding new material:

(4) A stranger arrived.

# An infinite number of sentences

1. Arbitrary long sentences can be built by adding new material:

(4) A tall stranger arrived.

## An infinite number of sentences

1. Arbitrary long sentences can be built by adding new material:

(4) A tall handsome stranger arrived.

## An infinite number of sentences

1. Arbitrary long sentences can be built by adding new material:

(4) A dark tall handsome stranger arrived.

# An infinite number of sentences

1. Arbitrary long sentences can be built by adding new material:

(4) A dark tall handsome stranger arrived suddenly.

# An infinite number of sentences

1. Arbitrary long sentences can be built by adding new material:

   (4) A dark tall handsome stranger arrived suddenly.

2. More interestingly, arbitrary long sentences can be built through center-embedding. In this case, there is a dependancy between arbitrary far apart elements:

   (5) The cats hunt.

   *center-embedding*: embedding a phrase in the middle of another phrase of the same type

## An infinite number of sentences

1. Arbitrary long sentences can be built by adding new material:

   (4) A dark tall handsome stranger arrived suddenly.

2. More interestingly, arbitrary long sentences can be built through center-embedding. In this case, there is a dependancy between arbitrary far apart elements:

   (5) The cats the neighbor owns hunt.

   *center-embedding*: embedding a phrase in the middle of another phrase of the same type

# An infinite number of sentences

1. Arbitrary long sentences can be built by adding new material:

   (4) A dark tall handsome stranger arrived suddenly.

2. More interestingly, arbitrary long sentences can be built through center-embedding. In this case, there is a dependancy between arbitrary far apart elements:

   (5) The cats the neighbor who arrived owns hunt.

   *center-embedding*: embedding a phrase in the middle of another phrase of the same type

# An infinite number of sentences (cont'd)

Consider the 3 structures:

- If $S_1$, then $S_2$.
- Either $S_1$ or $S_2$.
- The man who said $S_1$ is coming today.

1. The colored items are *dependent* one from the other
2. It is possible to create nested sentences of arbitrary length:

(6) If either the man who said $S_a$ is coming today, or $S_b$, then $S_c$.

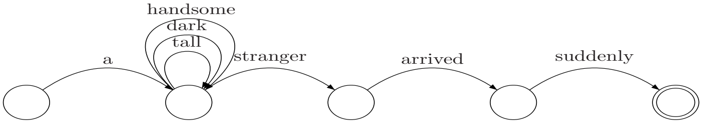
⇒ A look at various ways to form infinite sentences gives access to complexity.

# Overview

1. Formal Languages
2. Formal Grammars
3. Regular Languages
4. Formal complexity of Natural Languages
   - Introduction
   - Are NL regular?
   - Are NL context-free?
   - Are NL context-sensitive?

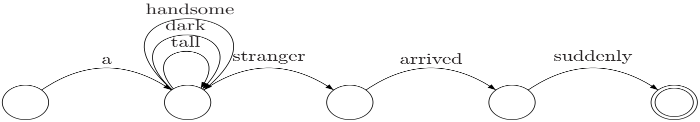
# Preliminaries: a word on lexicon

(7) A dark tall handsome stranger arrived suddently.



# Preliminaries: a word on lexicon

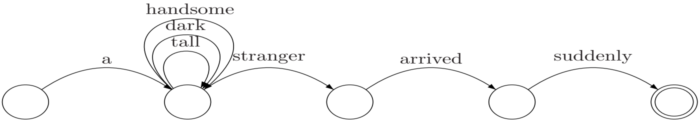
(7) A dark tall handsome stranger arrived suddently.



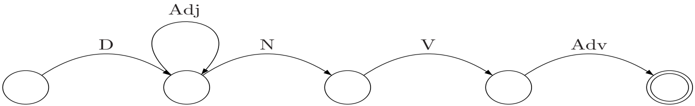

Let's leave aside lexicon issues

# Preliminaries: a word on lexicon

(7) A dark tall handsome stranger arrived suddently.




Let's leave aside lexicon issues



# Chomsky's first attempt

Consider the 3 structures:

- If $S_1$, then $S_2$.
- Either $S_1$ or $S_2$.
- The man who said $S_1$ is coming today.

1. The colored items are *dependent* one from the other
2. It is possible to create nested sentences of arbitrary length:

(8) If either the man who said $S_a$ is coming today, or $S_b$, then $S_c$.

Since such sentences are instances of mirroring and since the mirror language is not regular, then English is not regular (Chomsky, 1957, p. 22).

Fallacious claim: **a regular language may contain a non regular sub-language**

# Classical argument I

Let's consider the sentence(s):

(9) A man fired another man.

# Classical argument I

Let's consider the sentence(s):

(9) A man that a man hired fired another man.

# Classical argument I

Let's consider the sentence(s):

(9) A man that a man that a man hired hired fired another man.

# Classical argument I

Let's consider the sentence(s):

(9) A man that a man that a man hired hired fired another man.
A man (that a man)$^2$ (hired)$^2$ fired another man.

# Classical argument I

Let's consider the sentence(s):

(9) A man that a man that a man hired hired fired another man.
A man (that a man)$^2$ (hired)$^2$ fired another man.

The sentences (10) are all well-formed sentences (for any $n$).

(10) A man (that a man)$^n$ (hired)$^n$ fired another man.

# Classical Argument II

Let $x$ = that a man
$y$ = hired
$w$ = a man
$v$ = fired another man

- $wx^*y^*v$ is regular
- English $\cap\ wx^*y^*v = wx^ny^nv$ (10)
- If English is regular, then $wx^ny^nv$ must be regular (for the intersection of two regular languages is regular)
- But $wx^ny^nv$ is not regular (pumping lemma).
  Contradiction $\Rightarrow$ English is not regular.

(Schieber, 1985)

# Discussion

Counter arguments :

- Natural languages are finite
  - productivity doesn't seem to be bound
  - a list of all possible sentences, supposedly finite, is still too long for a human to learn
- People are bad at interpreting embedding: there might be a limit
  - there are indeed constraints on performance,
  - but in writing, or with an appropriate intonation, there doesn't seem to be a hard-wired limit

# Discussion: processing problems with nested structures

Psycholinguistic evidence that (11b) is more accepted than (11a) (Fodor, Frazier)

(11) a. The patient who the nurse who the clinic had hired admitted met Jack.
b. The patient who the nurse who the clinic had hired met Jack.

Other factors:

(12) a. The pictures which the photographer who I met yesterday took were damaged by the child.
b. ?The pictures which the photographer who John met yesterday took were damaged by the child.

(13) a. Isn't it true that example sentences [ that people [ that you know ] produce ] are more likely to be accepted? (De Roeck et al, 1982)
b. A book [ that some Italian [ I've never heard of ] wrote ] will be published soon by MIT Press (Frank, 1992)

*(Gibson & Thomas, 1997)*

# Examples

Bad examples :

(14) A girl that the man that the doctor knows like was fired.

Good examples:

(15) A foreman that an employee who were recently hired talked with was fired.

# Overview

1. Formal Languages
2. Formal Grammars
3. Regular Languages
4. Formal complexity of Natural Languages
   - Introduction
   - Are NL regular?
   - Are NL context-free?
   - Are NL context-sensitive?

# Pumping lemma: intuition

1. If a word is long enough, then there is (at least) one non terminal symbol appearing several times in its derivation.

"long enough" ?

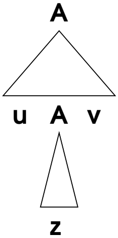
$$\begin{array}{lcl} S & \rightarrow & A\ B \\ A & \rightarrow & abaccabca \\ & | & abSba \\ B & \rightarrow & ccccc \end{array}$$

Minimal length : 14:

$S \rightarrow AB \rightarrow abaccabcaB \rightarrow abaccabcaccccc$

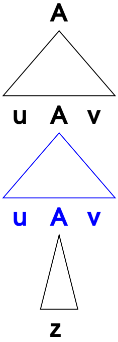
# Pumping lemma: intuition

2 Let's call this non terminal symbol *A*.



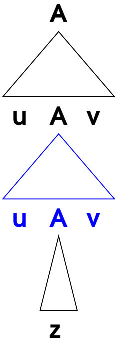
# Pumping lemma: intuition

2 Let's call this non terminal symbol $A$.



# Pumping lemma: intuition

2 Let's call this non terminal symbol $A$.




$$A \xrightarrow{*} uAv$$
$$A \xrightarrow{*} uAv \xrightarrow{*} uzv$$
$$A \xrightarrow{*} uAv \xrightarrow{*} uuAvv \xrightarrow{*} \underbrace{u \ldots u}_{n}\, z\, \underbrace{v \ldots v}_{n}$$

# Pumping Lemma for CF languages

**Def. 20 (Star lemma – CF languages)**

If $L$ is context-free, there exists $p \in \mathbb{N}$ such that:
$\forall w$ s.t. $|w| \geqslant p$,
$w$ can be factorized $w = rstuv$,
with:

$$|su| \geqslant 1$$
$$|stu| \leqslant p$$
$$\forall i \geqslant 0, \quad rs^i tu^i v \in L$$

(Bar-Hillel *et al.* , 1961)

# Pumping lemma: Consequences

The pumping lemma gives us a tool to prove that a language is **not** context-free.

| | | |
|---|---|---|
| $\mathcal{L}$ context-free | $\Rightarrow$ | pumping lemma $(\forall i, rs^i tu^i v \in \mathcal{L})$ |
| pumping lemma | $\not\Rightarrow$ | $\mathcal{L}$ context-free |

to prove that $\mathcal{L}$ is

context-free provide a type 2 grammar

not context-free show that the pumping lemma does not apply

# Results: expressivity

- well-parenthetized words (dyck's language) is context-free
  $S \rightarrow (S)S \mid \varepsilon$
- $a^n b^n (n \geqslant 0)$ is a context-free language
  $S \rightarrow aSb \mid \varepsilon$
- $ww^R, w \in \Sigma^*$ (mirror language) is a context-free language
  $S \rightarrow aSa \mid bSb \mid \varepsilon$
- $ww, w \in \Sigma^*$ (copy language) is not context-free
  proof: pumping lemma
- $a^n b^n c^n$ is not context-free
  proof: pumping lemma
- $a^m b^n c^m d^n$ is not context-free
  proof: pumping lemma
- $xa^m b^n yc^m d^n z$ is not context-free
  proof: pumping lemma

# Closure properties I

- CF languages are closed under rational operations
- union (gather all the rules, avoiding name conflicts, and adding a new start rule $S \rightarrow S_1 | S_2$),
- product ($S \rightarrow S_1 S_2$),
- and Kleene star ($S \rightarrow S_1 S \mid \varepsilon$).

# Closure properties II : intersection

- CF languages are not closed under intersection

**Example**

$L_1 = \{a^i b^i c^j \mid i, j \geq 0\}$ is context-free:
$$S \rightarrow XY$$
$$X \rightarrow aXb \mid \varepsilon$$
$$Y \rightarrow cY \mid \varepsilon$$

$L_2 = \{a^i b^j c^j \mid i, j \geq 0\}$ is also context-free:
$$S \rightarrow XY$$
$$X \rightarrow aX \mid \varepsilon$$
$$Y \rightarrow bYc \mid \varepsilon$$

But $L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 0\}$ is not contex-free.

# Closure properties III: other results

- CF languages are not closed under complement (since they are not closed under intersection)
- CF languages are closed under intersection with a regular language
- a sub-class of CF languages, *deterministic CF languages* are closed for set complement, but not for union (one can easily define an intrinsequely non deterministic language as the union of two “independant” languages)

# Final argument I

After many attempts by various scholars, attempts which are severely critized and ruined in (Gazdar & Pullum, 1985), Schieber (1985) came up with a widely accepted answer:

1. In swiss-german, subordinate clauses can have a structure where all NPs precede all Vs. (16)

(16) Jan säit das mer NP* es huus haend wele V* aastrüche
Jan said that we NP* the house have wanted V* paint
'Jan said that we have wanted (that) V* NP* paint the house'

2. Among those subordinate clauses, those where all the dative NPs precede all the accusative NPs are well-formed. (17)

(17) … das mer d'chind em Hans es huus haend wele laa hälfe aastrüche
… that we the_children.ACC Hans.DAT the house.ACC have wanted let help paint
'… that we have wanted to let the children help Hans to paint the house'

# Final argument II

3. The number of verbs requiring a dative has to be equal to the number of dative NPs, the same for accusative.
4. The number of verbs in a subordinate clause is limited only by performance

Let $R$ be the language:

R $=$ {Jan säit das mer (d'chind)$^h$ (em Hans)$^i$ es huus haend wele (laa)$^j$ (hälfe)$^k$ aastrüche, $i, j, k, h \geqslant 1$}

Then let $L =$ Swiss-German $\cap$ $R$ $=$

{Jan säit das mer (d'chind)$^m$ (em Hans)$^n$ es huus haend wele (laa)$^m$ (hälfe)$^n$ aastrüche, $m, n \geqslant 1$}

$L$ is not context-free, whereas $R$ is regular.

$\Rightarrow$ Swiss-German is not context-free.

# Overview

1. Formal Languages
2. Formal Grammars
3. Regular Languages
4. Formal complexity of Natural Languages
   - Introduction
   - Are NL regular?
   - Are NL context-free?
   - Are NL context-sensitive?

# Current proposal

1. The context-sensitive class seems too big: for instance $\{a^{2^i} \;/\; i \geqslant 0\}$ is context-sensitive.
2. Joshi (1985) proposed a subclass of type 1 languages, namely the class of *mildly context-sensitive languages* (MCSL), this class has the following properties:
   - $ww$ is MCS
   - $a^n b^n c^n$ is MCS
   - $a^n b^n c^n d^n$ is MCS
   - $a^i b^j c^i d^j$ is MCS
   - $a^n b^n c^n d^n e^n$ is not MCS
   - $www$ is not MCS
   - $ab^h ab^i ab^j ab^k ab^l, h > i > j > k > l \geqslant 1$ is not MCS
   - $a^{2^i}$ is not MCS

# Current proposal

1. The context-sensitive class seems too big: for instance $\{a^{2^i} \ / \ i \geqslant 0\}$ is context-sensitive.
2. Joshi (1985) proposed a subclass of type 1 languages, namely the class of *mildly context-sensitive languages* (MCSL), this class has the following properties:
   - $ww$ is MCS
   - $a^n b^n c^n$ is MCS
   - $a^n b^n c^n d^n$ is MCS
   - $a^i b^j c^i d^j$ is MCS
   - $a^n b^n c^n d^n e^n$ is not MCS
   - $www$ is not MCS
   - $ab^h ab^i ab^j ab^k ab^l, h > i > j > k > l \geqslant 1$ is not MCS
   - $a^{2^i}$ is not MCS

Conjecture : NL $\in$ MCSL

# More about MCSL

Interesting properties of MCSL:

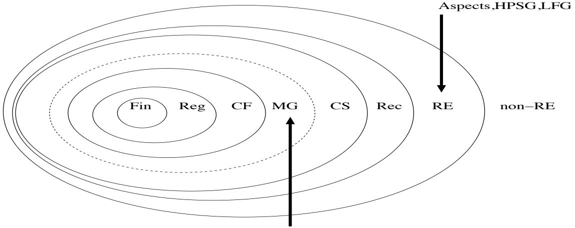
- restricted growth: if $L$ is MCS, there is $k$ such that for all words $w \in L$, there is a word $w'$ s.t. $|w'| \leqslant |w| + k$
- word problem for MCSL are of a polynomial complexity

These properties are arguably common with natural languages

The formalism introduced by Joshi, *Tree Adjoining Grammars*, defines the class of MCSL.

# Minimalist grammars (Stabler, 2011)

Minimalist grammars (MGs), as defined here by (5), (6) and (8), have been studied rather carefully. It has been demonstrated that the class of languages definable by minimalist grammars is exactly the class definable by multiple context free grammars (MCFGs), linear context free rewrite systems (LCFRSs), and other formalisms [62,64,66,41]. MGs contrast in this respect with some other much more powerful grammatical formalisms (notably, the 'Aspects' grammar studied by Peters and Ritchie [76], and HPSG and LFG [5,46,101]):




The MG definable languages include all the finite (Fin), regular (Reg), and context free languages (CF), and are properly included in the context sensitive (CS), recursive (Rec), and recursively enumerable languages (RE). Languages definable by tree adjoining grammar (TAG) and by a certain categorial combinatory grammar (CCG) were shown by Vijay Shanker and Weir to be sandwiched inside the MG class [103].[4] With all these results

**Theorem 1.** $CF \subset \boxed{TAG \equiv CCG} \subset \boxed{MCFG \equiv LCFRS \equiv MG} \subset CS.$

# References I

Bar-Hillel, Yehoshua, Perles, Micha, & Shamir, Eliahu. 1961. On formal properties of simple phrase structure grammars. *STUF-Language Typology and Universals*, 14(1-4), 143–172.

Chomsky, Noam. 1957. *Syntactic Structures*. Den Haag: Mouton & Co.

Gazdar, Gerald, & Pullum, Geoffrey K. 1985 (May). *Computationally Relevant Properties of Natural Languages and Their Grammars*. Tech. rept. Center for the Study of Language and Information, Leland Stanford Junior University.

Gibson, Edward, & Thomas, James. 1997. The Complexity of Nested Structures in English: Evidence for the Syntactic Prediction Locality Theory of Linguistic Complexity. *Unpublished manuscript, Massachusetts Institute of Technology*.

Joshi, Aravind K. 1985. *Tree Adjoining Grammars: How Much Context-Sensitivity is Required to Provide Reasonable Structural Descriptions?* Tech. rept. Department of Computer and Information Science, University of Pennsylvania.

Langendoen, D Terence, & Postal, Paul Martin. 1984. *The vastness of natural languages*. Basil Blackwell Oxford.

Mannell, Robert. 1999. *Infinite number of sentences*. part of a set of class notes on the Internet. http://clas.mq.edu.au/speech/infinite_sentences/.

Schieber, Stuart M. 1985. Evidence against the Context-Freeness of Natural Language. *Linguistics and Philosophy*, 8(3), 333–343.

Stabler, Edward P. 2011. Computational perspectives on minimalism. *Oxford handbook of linguistic minimalism*, 617–643.