Complexity and Character of Human Languages

Informatics 2A: Lecture 25

John Longley

15 November 2012

1. Human Language Complexity
   - Chomsky Hierarchy
   - The Faculty of Language
   - Strong and Weak Adequacy

2. Linear Indexed Grammars

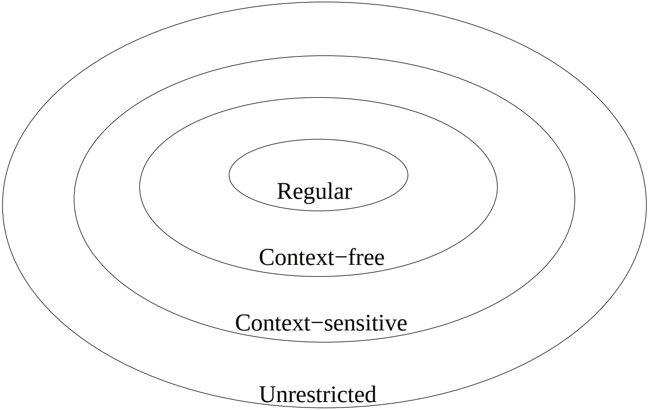
Reading: J&M. Chapter 16.3–16.4.

Review

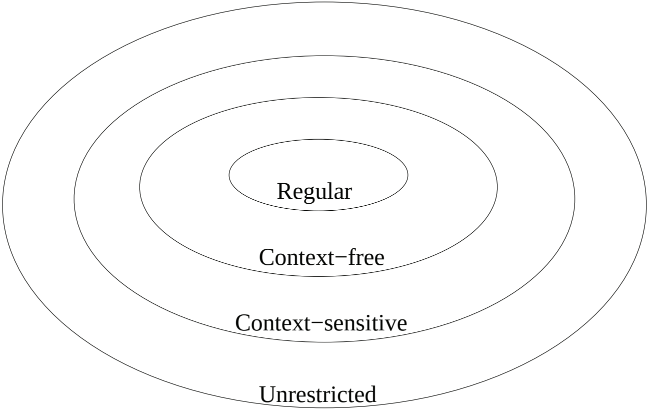
Chomsky Hierarchy: classifies languages on scale of complexity:

- Regular languages: those whose phrases can be 'recognized' by a finite state machine.
- Context-free languages: the set of languages accepted by pushdown automata. Many aspects of PLs and NLs can be described at this level;
- Context-sensitive languages: equivalent with a linear bounded nondeterministic Turing machine, also called a linear bounded automaton. Need this to capture e.g. *typing rules* in PLs.
- Unrestricted languages: *all* languages that can in principle be defined via mechanical rules.

Review

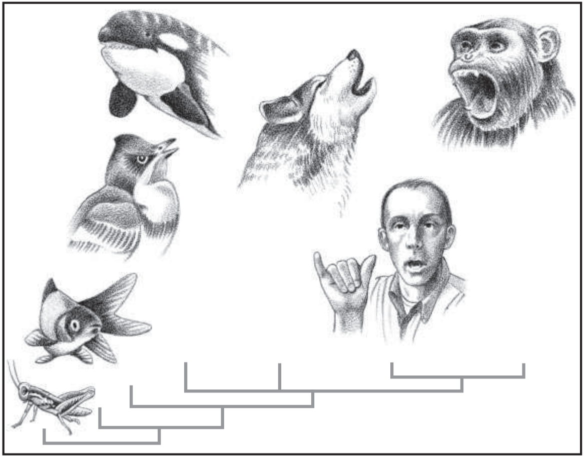


Review



Where do human languages fit within this complexity hierarchy?

The Faculty of Language



The "language faculty" has a broad sense and a narrow sense (Hauser, Chomsky, and Fitch 2002).

The Faculty of Language (Broad Sense)

Sensory-motor system

- for producing and perceiving linguistic communication
- spoken language: vocal track, auditory system
- sign language: gestural system, visual system
- written language: writing system, visual or tactile system

Conceptual-intentional system

- who to communicate with and what to communicate about
- generating mental states and attributing them to others;
- acquiring conceptual representations that are non-linguistic;
- referring to entities and events.

The Faculty of Language (Narrow Sense)

Abstract computational system

- one part of which is narrow syntax which generates internal representations and ties them into:
- sensory-motor interface through phonological, gestural system;
- conceptual-intentional system through semantic (and pragmatic) systems.

A core property of narrow syntax is recursion: a finite set of 'rules' yields a potentially infinite set of discrete expressions.

Recursion

The potential infiniteness of the language faculty has been recognized by Galileo, Descartes, von Humboldt.

Discrete Infinity

- Sentences are built up by discrete units
- There are 6-word sentences, and 7-word sentences, but no 6.5 word sentences
- There is no longest sentence!
- There is no non-arbitrary upper bound to sentence length!

Mary thinks that John thinks that George thinks that Mary thinks that this course is boring!
I ate lunch and slept and watched tv and went to the bathroom and had a coffee and got dressed . . .

Strong and Weak Adequacy

Questions about the formal complexity of language are about the computational power of syntax, as represented by a grammar that's adequate for it.

A strongly adequate grammar

- generates all and only the strings of the language;
- assigns them the "right" structures — ones that support a correct representation of meaning. (See previous lecture.)

A weakly adequate grammar

generates all and only the strings of a language but doesn't necessarily give a correct (insightful) account of their structures.

Is Natural Language Regular?

It is generally agreed that NLs are not (in principle) regular!

Centre-embedding

[The cat_1 likes tuna fish_1].
[The cat_1 [the dog_2 chased_2] likes tuna fish_1].
[The cat_1 [the dog_2 [the rat_3 bit_3] chased_2] likes tuna fish_1].

Idea of proof

(the+noun)^n (transitive verb)$^{n-1}$ likes tuna fish.
A = { the cat, the dog, the rat, the elephant, the kangaroo ... }
B = { chased, bit, admired, ate, befriended ... }
Intersect `/A* B* likes tuna fish/` with English
$L = x^n y^{n-1}$ *likes tuna fish*, $x \in A, y \in B$
Use pumping lemma to show L is not regular

Is Natural Language Context Free?

It seems NLs aren't always context free! E.g. in Swiss German, some verbs (e.g. *let*, *paint*) take an object in accusative form, while others (e.g. *help*) take it in dative form.

Crossing dependencies

...das mer	d'chind	em Hans	es huus	lönd	hälfe	aastriiche
...that we	the children	Hans	the house	let	help	paint
	NP-ACC	NP-DAT	NP-ACC	V-ACC	V-DAT	V-ACC

...that we let the children help Hans paint the house

Abstracting out the key feature here, we see that the same sequence over $\{a, d\}$ (in this case *ada*) must 'appear twice'.

But it turns out that $\{ss \mid s \in \{a, d\}^*\}$ isn't context-free (see a later lecture). Hence neither is Swiss German!

Weaker examples

These ‘crossing dependencies’ are non-context-free in a very strong sense: no CFG is even weakly adequate for modelling them.

Other phenomena can *in theory* be modelled using CFGs, though it seems unnatural to do so. E.g. a versus an in English.

a banana	an apple
a large apple	an exceptionally large banana

Over-simplifying a bit: a before consonants, an before vowels.

In theory, we could use a context-free grammar:

$$\begin{array}{rcl} NP & \rightarrow & \textbf{a}\ NP1^c \\ NP1^c & \rightarrow & N^c \mid AP^c\ NP1 \\ AP^c & \rightarrow & A^c \mid Adv^c\ AP \end{array} \qquad \begin{array}{rcl} NP & \rightarrow & \textbf{an}\ NP1^v \\ NP1^v & \rightarrow & N^v \mid AP^v\ NP1 \\ AP^v & \rightarrow & A^v \mid Adv^v\ AP \end{array}$$

But more natural to use context-sensitive rules e.g.

$$\begin{array}{rcl} DET\ [\text{c-word}] & \rightarrow & \textbf{a}\ [\text{c-word}] \\ DET\ [\text{v-word}] & \rightarrow & \textbf{an}\ [\text{v-word}] \end{array}$$

Linear Indexed Grammars

Linear indexed grammars (LIGs) are more powerful than CFGs, but much less powerful than an arbitrary CSGs. Think of them as mildly context sensitive grammars.

Definition

An indexed grammar has three disjoint sets of symbols: terminals, non-terminals and indices.

An index is a stack of symbols that can be passed from the LHS of a rule to its RHS, allowing counting and recording what rules were applied in what order.

Linear Indexed Grammars (not examinable)

$\text{S} \rightarrow \text{D}_f$	pushes an f onto the index on D
$\text{D} \rightarrow \text{D}_g$	pushes a g onto the index on D
$\text{D} \rightarrow \text{ABC}$	passes the index on D to A, B and C

$\text{g} = \langle\ \text{A} \rightarrow \text{Aa} \mid \text{B} \rightarrow \text{Bb} \mid \text{C} \rightarrow \text{Cc}\ \rangle$	pops g from an index
$\text{f} = \langle\ \text{A} \rightarrow \text{a} \mid \text{B} \rightarrow \text{b} \mid \text{C} \rightarrow \text{c}\ \rangle$	pops f from an index

Derivation in an Indexed Grammar

$$\begin{array}{ll} S \rightarrow D_f & g = \langle\ A \rightarrow Aa \mid B \rightarrow Bb \mid C \rightarrow Cc\ \rangle \\ D \rightarrow D_g & f = \langle\ A \rightarrow a \mid B \rightarrow b \mid C \rightarrow c\ \rangle \\ D \rightarrow ABC & \end{array}$$

S

Derivation in an Indexed Grammar

$$\begin{array}{ll} \mathrm{S} \rightarrow \mathrm{D}_f & \mathrm{g} = \langle\ \mathrm{A} \rightarrow \mathrm{Aa} \mid \mathrm{B} \rightarrow \mathrm{Bb} \mid \mathrm{C} \rightarrow \mathrm{Cc}\ \rangle \\ \mathrm{D} \rightarrow \mathrm{D}_g & \mathrm{f} = \langle\ \mathrm{A} \rightarrow \mathrm{a} \mid \mathrm{B} \rightarrow \mathrm{b} \mid \mathrm{C} \rightarrow \mathrm{c}\ \rangle \\ \mathrm{D} \rightarrow \mathrm{ABC} & \end{array}$$

$$\begin{array}{c} \mathrm{S} \\ | \\ \mathrm{D}_f \end{array}$$

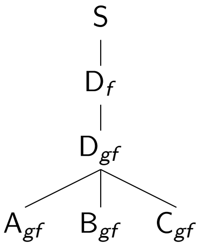
Derivation in an Indexed Grammar

$$
\begin{array}{ll}
S \rightarrow D_f & g = \langle\ A \rightarrow Aa \mid B \rightarrow Bb \mid C \rightarrow Cc\ \rangle \\
D \rightarrow D_g & f = \langle\ A \rightarrow a \mid B \rightarrow b \mid C \rightarrow c\ \rangle \\
D \rightarrow ABC &
\end{array}
$$

$$
\begin{array}{c}
S \\
| \\
D_f \\
| \\
D_{gf}
\end{array}
$$

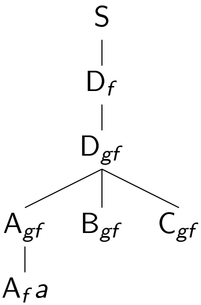
Derivation in an Indexed Grammar

$$
\begin{array}{ll}
S \rightarrow D_f & g = \langle\ A \rightarrow Aa \mid B \rightarrow Bb \mid C \rightarrow Cc\ \rangle \\
D \rightarrow D_g & f = \langle\ A \rightarrow a \mid B \rightarrow b \mid C \rightarrow c\ \rangle \\
D \rightarrow ABC &
\end{array}
$$



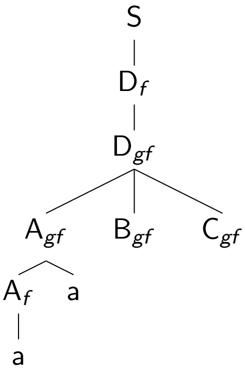
Derivation in an Indexed Grammar

$$\begin{array}{ll} S \rightarrow D_f & g = \langle\ A \rightarrow Aa \mid B \rightarrow Bb \mid C \rightarrow Cc\ \rangle \\ D \rightarrow D_g & f = \langle\ A \rightarrow a \mid B \rightarrow b \mid C \rightarrow c\ \rangle \\ D \rightarrow ABC & \end{array}$$



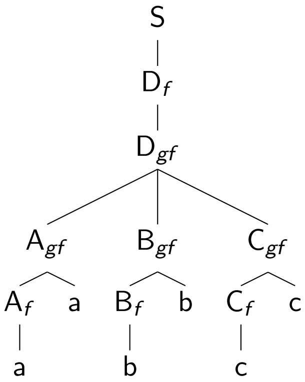
Derivation in an Indexed Grammar

$$
\begin{array}{ll}
S \to D_f & g = \langle\, A \to Aa \mid B \to Bb \mid C \to Cc \,\rangle \\
D \to D_g & f = \langle\, A \to a \mid B \to b \mid C \to c \,\rangle \\
D \to ABC &
\end{array}
$$



Derivation in an Indexed Grammar

$$
\begin{array}{ll}
S \rightarrow D_f & g = \langle\, A \rightarrow Aa \mid B \rightarrow Bb \mid C \rightarrow Cc \,\rangle \\
D \rightarrow D_g & f = \langle\, A \rightarrow a \mid B \rightarrow b \mid C \rightarrow c \,\rangle \\
D \rightarrow ABC &
\end{array}
$$



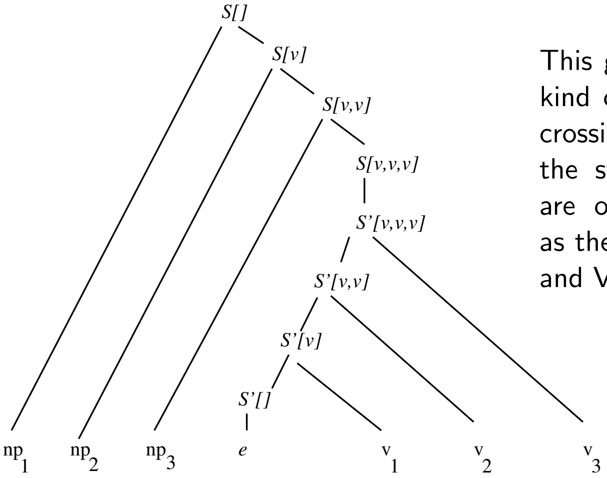
Linear Indexed Grammars

Linear Indexed Grammars (LIGs) allow an index to pass to only one non-terminal on the RHS (not three, as in previous example).

Here we'll push numbers onto an index.
An LIG for crossing dependencies in $np^k v^k$:

$S_{[\ldots]}$	$\rightarrow$	$np_i \; S_{[i,\ldots]}$	emit NP, push a number
$S_{[\ldots]}$	$\rightarrow$	$S'_{[\ldots]}$	switch to verb sequence rule
$S'_{[i,\ldots]}$	$\rightarrow$	$S'_{[\ldots]} \; v_i$	pop a number, emit a verb
$S'_{[\,]}$	$\rightarrow$	ϵ	stop if stack is empty

Example: LIG derivation for np^3v^3



This grammar produces the kind of strings we want for crossing dependencies, but the structures it generates are only weakly adequate, as they don't associate NPs and Vs directly.

Linear Indexed Grammars

In view of the weak adequacy of LIGs, other 'mildly context-sensitive' grammar formalisms have been developed that are strongly adequate for NL:

- Tree Adjoining Grammar (TAG): a system of tree re-writing rules (ie, not string re-writing rules) in which elementary trees are combined by substitution and adjunction;
- Combinatory Categorial Grammar (CCG): a system that links words to complex categories that specify how adjacent words fit together, in terms of combinators like apply a functor to an argument, compose two functors, etc..

Summary

- The 'narrow' language faculty involves a computational system that generates syntactic representations that can be mapped onto meanings.
- This raises the question of the complexity of this system (its position in the Chomsky hierarchy).
- A weakly adequate grammar generates the correct strings, while a strongly adequate one also generates the correct structures.
- NLs appear to surpass the power of context-free languages, but only just.
- The mild form of context-sensitivity captured by LIGs seems weakly adequate for NL structures.

Next Lecture: Models of human parsing.