

Languages

Definition: An *alphabet* is a finite, nonempty set of symbols. We use Σ to denote this alphabet.

A *string* is a finite sequence of symbols from Σ .

The length of a string s , denoted $|s|$, is the number of symbols in it.

The empty string, ϵ , is the string of length zero.

Σ^* denotes the set of all sequences of strings that are composed of zero or more symbols of Σ .

Σ^+ denotes the set of all sequences of strings composed of one or more symbols of Σ , i.e. $\Sigma^* - \{\epsilon\}$.

A *language* is a subset of Σ^* .

Grammars

Defined as a quadruple $\langle V_t, V_n, P, S \rangle$, where:

- V_t = the *terminal vocabulary*, i.e. the words of the language: finite (but productive ...)
- V_n = the *non-terminals*, i.e. the set of categories, used for the benefit of the grammar to express generalizations over items in V_t
- P = the *set of productions*, i.e. rules in the grammar: finite, and of the form $\alpha \rightarrow \beta$, where $\alpha, \beta \in (V_n \cup V_t)$
- S = the *distinguished symbol*, or 'start' symbol (cf. 'sentence', if we're talking natural languages):
 - $S \in V_n$,
 - P must include at least one rule $\alpha \rightarrow \beta$ where $\alpha = S$.

A grammar is *generative* if it predicts (explicitly defines, or characterizes) *all and only all* strings \in the language.

Chomsky Hierarchy

There are different types of languages, and grammars whose *power* depends on the nature of $\alpha, \beta \in P$.

Type	Grammar	Language	Automata
3	Finite State	Regular	Finite
2	Context-Free	C-F	Pushdown
1	Context-Sensitive	C-S	Linear-Bounded
0	General Rewrite	Unrestricted	Turing Machines

Type 3 \subset Type 2 \subset Type 1 \subset Type 0

Type 3: Finite State

$\alpha \rightarrow \beta$

- α is a single non-terminal ($\in V_n$)
- β is either:
 - a single terminal ($\in V_t$), or
 - a string with a single terminal and a single non-terminal

e.g. $A \rightarrow a$, or $A \rightarrow aA$ (or $A \rightarrow Aa$).

Note that this allows for recursion, and so can handle (some) languages containing an infinite number of strings.

Question : what's this language called?

Question : rewrite the grammar in formal terms.

Question : rewrite the grammar in terms of a FS-machine. Is there more than one way of doing this? Can we compare these alternatives?

Type 3: Finite State

Another (more interesting?) example:

$$\langle \begin{array}{l} V_t = \{walk, eat, sleep, er, s, chair, table\} \\ V_n = \{PN, N, V\} \\ P = \{PN \rightarrow N \ s, N \rightarrow V \ er, \\ N \rightarrow chair, N \rightarrow table, \\ V \rightarrow walk, V \rightarrow sleep, V \rightarrow eat\} \\ S = PN \rangle \end{array}$$

Check all these rules are *bona fide* FS!

Question : what's the complete language that this grammar defines?

Question : rewrite this grammar in terms of a FSM. Does it capture generalizations? If not, is there a way in which we can capture them?

Note: allows left- or right-branching only!!

Type 2: Context Free

$\alpha \rightarrow \beta$

- α is a single non-terminal ($\in V_n$)
- β is a string of terminals and non-terminals

$$\langle \begin{array}{l} V_t = \{the, cat, dog, mouse, eats\} \\ V_n = \{S, NP, VP, D, N, V\} \\ P = \{S \rightarrow NP, VP \\ NP \rightarrow D, N \\ VP \rightarrow V \\ VP \rightarrow V, NP \end{array}$$

$D \rightarrow th$
 $N \rightarrow c$
 $N \rightarrow d$
 $N \rightarrow m$
 $V \rightarrow e$

$$S = S \rangle$$

Question : what's the complete language that this grammar defines?

Question : which of these rules could be part of a FSG? Which ones make this a CFG?

i.e. Type 3 \subset Type 2.

Type 2: Context Free

Note that CFGs allow for centre-embedding:

$$S \rightarrow a, S, b$$

$$S \rightarrow a, b$$

Question : what are the first 5 strings (by length) in this language? What is this language called? Draw the structures of the three shortest strings.

Note the recursion!

Question : rewrite the grammar in formal terms.

Question : show that this grammar cannot be rewritten in terms of a FS-machine.

Type 1: Context Sensitive

$$\alpha \rightarrow \beta$$

- α and β are strings of terminals and non-terminals ($\in (V_n \cup V_t)$), but β must be no shorter than α

$$\langle \begin{array}{l} V_t = \{the, cat, cats, dog, dogs, run, runs\} \\ V_n = \{S, V\} \\ P = \{S \rightarrow the\ cat\ V \quad S \rightarrow the\ cats\ V \\ the\ cat\ V \rightarrow the\ cat\ runs \\ the\ cat\ V \rightarrow the\ dog\ runs \\ the\ cats\ V \rightarrow the\ dogs\ runs \\ the\ cats\ V \rightarrow the\ cats\ runs\} \end{array} \rangle$$

Question : what's the complete language that this grammar defines?

Question : which of these rules could be part of a CFG? FSG?

i.e. Type 2 \subset Type 1.

Note, $ABC \rightarrow AEDC$ can draw a tree; impossible with $ABC \rightarrow DEFG$.

Type 1: Context Sensitive

Assuming the ruleset:

- $S \rightarrow abc$
- $S \rightarrow aSBc$
- $cB \rightarrow Bc$
- $bB \rightarrow bb$

Question : what are the first three strings in this language? What is this language called? Show their derivations (i) using trees; (ii) using string manipulation. Does it make a difference in which order you apply the rules?

Question : rewrite the grammar in formal terms.

Type 0: General Rewrite Systems

$\alpha \rightarrow \beta$

- α and β are strings of terminals and non-terminals
($\in (V_n \cup V_t)$)

\langle $V_t =$ {*the, book, John, bought, is, by, was,*
 $V_n =$ {*S, NP, VP, V, D, N*}
 $P =$ { $S \rightarrow NP, VP$
 $NP \rightarrow D, N$
 $VP \rightarrow V, NP$

John bought the book \rightarrow *the book was*
bought by John
The book bought John \rightarrow *John is sitting*
 $S = S$

Question : what's the complete language that this grammar defines?

Question : which of these rules could be part of a FSG? Which ones make this a CFG? CSG?

Summary

FS	most restrictive least powerful	↓	GRS	least restrictive most powerful	makes strong claim	makes weak claim
...		↓	...			

Note also the difference between:

- *weak generative capacity*: whether a grammar characterizes a string as a member of a language or not;
- *strong generative capacity*: whether a grammar also gives the structure for that string

Context-Free Grammars are *guaranteed* to have strong generative capacity.