GENERATIVE CAPACITY

Grammatical formalisms can be classified by their **generative capacity**: the set of languages they can represent. Chomsky (1957) describes four classes of grammatical formalisms that differ only in the form of the rewrite rules. The classes can be arranged in a hierarchy, where each class can be used to describe all the languages that can be described by a less powerful class, as well as some additional languages. Here we list the hierarchy, most powerful class first:

Recursively enumerable grammars use unrestricted rules: both sides of the rewrite rules can have any number of terminal and nonterminal symbols. These grammars are equivalent to Turing machines in their expressive power.

Context-sensitive grammars are restricted only in that the right-hand side must contain at least as many symbols as the left-hand side. The name context-sensitive comes from the fact that a rule such as $ASB \rightarrow AXB$ says that an S can be rewritten as an X in the context of a preceding A and a following B.

In **context-free grammars** (or **CFG**s), the left-hand side consists of a single nonterminal symbol. Thus, each rule licenses rewriting the nonterminal as the right-hand side in *any* context. CFGs are popular for natural language grammars, although it is now widely accepted that at least some natural languages are not context-free (Pullum, 1991).

Regular grammars are the most restricted class. Every rule has a single nonterminal on the left-hand side, and a terminal symbol optionally followed by a nonterminal on the right-hand side. Regular grammars are equivalent in power to finite-state machines. They are poorly suited for programming languages because, for example, they cannot represent constructs such as balanced opening and closing parentheses.

To give you an idea of which languages can be handled by which classes, the language $a^n b^n$ (a sequence of n copies of a followed by the same number of b) can be generated by a context-free grammar, but not a regular grammar. The language $a^n b^n c^n$ requires a context-sensitive grammar, whereas the language $a^* b^*$ (a sequence of any number of a followed by any number of b) can be described by any of the four classes. A summary of the four classes follows.

Class	Sample Rule	Sample Language
Recursively enumerable	$A\ B \rightarrow C$	any
Context-sensitive	$A\ B \rightarrow B\ A$	$a^n b^n c^n$
Context-free	$S \rightarrow \boldsymbol{a}\ S\ \boldsymbol{b}$	$a^n b^n$
Regular	$S \rightarrow \boldsymbol{a}\ S$	$a^* b^*$