Chomsky hierarchy

Hierarchy of grammars according to Chomsky is explained below as per the grammar types −

Type 0. Unrestricted grammars

Turing Machine (TM)

Type 1. Context-sensitive grammars

Linear Bounded Automaton (LBA)

Type 2. Context-free grammars

Pushdown Automaton (PDA)

Type 3. Regular grammars

Finite Automaton (FA)

Type-0 unrestricted grammar

- Type-0 grammars generate recursively enumerable.
- In type-0 the production has no restrictions.
- There may be any phase structure grammar which includes all formal grammars
- They generate the language which is recognized by the Turing machine.
- The productions can be in the form of a->b where, a is a string of terminals and with at least one non-terminals and a cannot be null. b is a string of terminal and non-terminal.

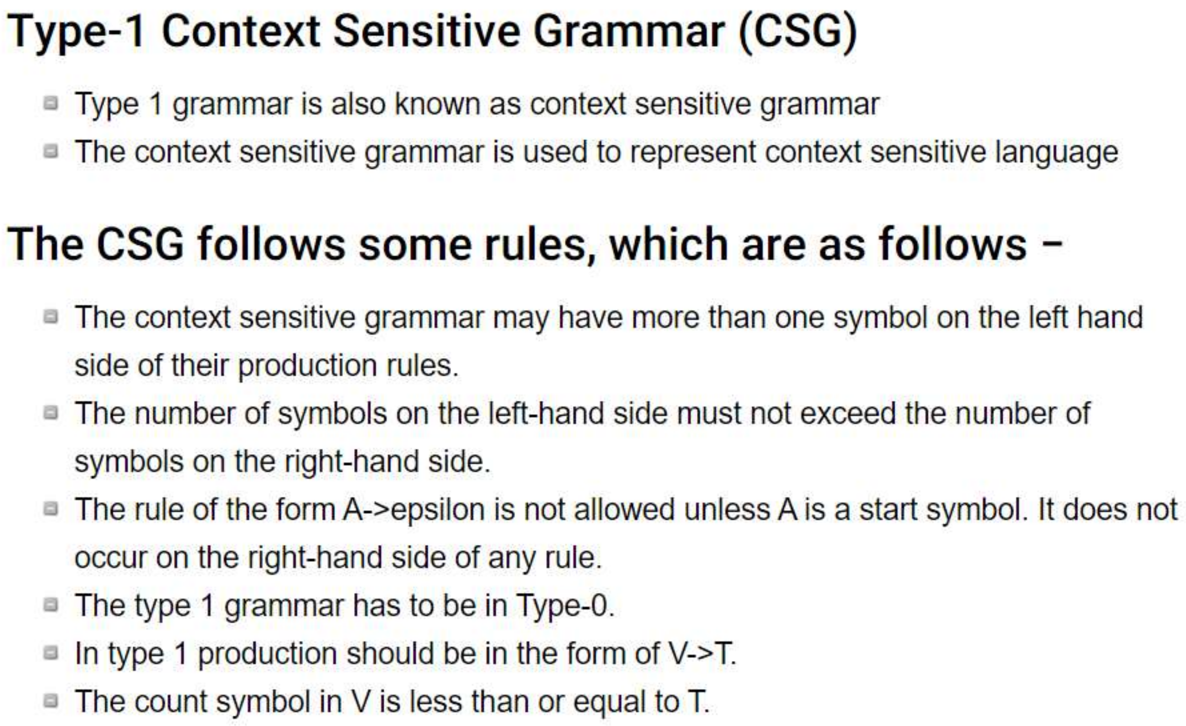
Example

S->ACaB

Bc->acB

CB->DB

aD->Db



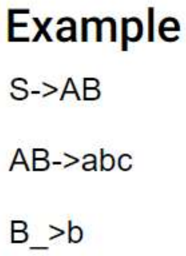
Type-1 Context Sensitive Grammar (CSG)

- Type 1 grammar is also known as context sensitive grammar
- The context sensitive grammar is used to represent context sensitive language

The CSG follows some rules, which are as follows –

- The context sensitive grammar may have more than one symbol on the left hand side of their production rules.
- The number of symbols on the left-hand side must not exceed the number of symbols on the right-hand side.
- The rule of the form A->epsilon is not allowed unless A is a start symbol. It does not occur on the right-hand side of any rule.
- The type 1 grammar has to be in Type-0.
- In type 1 production should be in the form of V->T.
- The count symbol in V is less than or equal to T.

***(A non-terminal may have an epsilon production rules if that non-terminal does not appear on the right-hand side of any production rule.)



Example

S->AB

AB->abc

B_>b

Type 2 – Context Free Grammar (CFG)

- Type 2 grammars are generated by context free languages.
- The language that is generated by the grammar is recognized by Push Down Automata.
- Type 2 must be in Type 1.
- Left-hand side of production can have only one variable.
- |alpha| =1

  There is no restriction on beta.

  The production rules are in the form of −

  A->alpha

  Where, A is any single non-terminal and is any combination of terminals and nonterminals.

Example

S->AB

A->a

B->b

Type 3 – Regular grammar

- Type 3 grammars are generated by regular languages.
- These languages are exactly all those languages that can be accepted by finite state automata.
- Type 3 is the most restricted grammar.
- It is in the form of

  V->VT*/T* Or V->T*V/T*

Example

S->ab