• **Definition**

  A *grammar* is a quadruple \( G = (V, \Sigma, S, R) \), where

  \( V \) is a finite set of *variables*,
  \( \Sigma \) is a finite set of *terminals* (disjoint from \( V \)),
  \( S \) is an element of \( V \), called the *start variable*, and
  \( R \) is a finite set of *rules* \( u \rightarrow v \), where \( u, v \in (V \cup \Sigma)^* \) and \( u \neq \epsilon \).

• Rules are also called *productions*.

• The *Chomsky hierarchy* classifies grammars according to syntactic restrictions on rules.
Chomsky Hierarchy

- **Regular grammars**
  Rules are of the form $A \rightarrow bC$ or $A \rightarrow a$, where $A$, $B$, and $C$ are variables and $a$ and $b$ terminals.

- **Context-free grammars**
  Rules are of the form $A \rightarrow v$, where $A$ is a variable and $v$ a string of variables and/or terminals.

- **Context-sensitive grammars**
  Rules are of the form $uAw \rightarrow uvw$, where $A$ is a variable, and $u$ and $w$ are arbitrary strings, and $v$ a non-empty string, of variables and/or terminals.

- **Unrestricted grammars**
  Rules are of the form $u \rightarrow v$, where $u$ and $w$ are strings of variables and/or terminals.
Derivations

- We write $s \Rightarrow t$ if the string $t$ can be obtained from $s$ by replacing a substring that matches the left-hand side of a rule by the corresponding right-hand side.

- That is, we have $s \Rightarrow t$, if there exist a rule $u \rightarrow v$ and strings $x$ and $y$ such that $s = xuy$ and $t = xvy$.

- A derivation is a sequence of substring replacement steps,
  
  $$u_0 \Rightarrow u_1 \Rightarrow \cdots \Rightarrow u_n.$$ 

- We also say that $u_n$ can be derived from $u_0$ in $G$, and write $u_0 \Rightarrow^* u_n$.

- The language $L(G)$ generated by $G$ is the set
  
  $$\{w \in \Sigma^* : S \Rightarrow^* w\},$$

  of all strings of terminals that can be derived from the start variable.
Example

- Let $G = (V, \Sigma, S, R)$ be a grammar, where

\[
\begin{align*}
V &= \{S, A, B, C\}, \\
\Sigma &= \{a, b, c\}, \text{ and} \\
R &= \{S \rightarrow \epsilon, \\
&\quad S \rightarrow ABCS, \\
&\quad AB \rightarrow BA, \\
&\quad BA \rightarrow AB, \\
&\quad AC \rightarrow CA, \\
&\quad CA \rightarrow AC, \\
&\quad BC \rightarrow CB, \\
&\quad CB \rightarrow BC, \\
&\quad A \rightarrow a, \\
&\quad B \rightarrow b, \\
&\quad C \rightarrow c\}
\end{align*}
\]

- What language is generated by this grammar?
Example

• We next modify the grammar $G$ to a grammar $G_1 = (V_1, \Sigma, S, R_1)$, where

\[
V_1 = \{S, A, B, C\} \quad \text{and} \\
R_1 = \{S \rightarrow aSBC, \\
S \rightarrow \epsilon, \\
CB \rightarrow BC, \\
aB \rightarrow ab, \\
bB \rightarrow bb, \\
bC \rightarrow bc, \\
cC \rightarrow cc\}
\]

• This grammar generates $\{a^n b^n c^n : n \geq 0\}$. 

Example

- The grammar $G_2 = (V_2, \Sigma_2, S, R_2)$, where

  $V_2 = \{S, D, R, T, [], \}$

  $\Sigma_2 = \{a\}$, and

  $R_2 = \{S \rightarrow [Da],$
  
  $S \rightarrow a,$

  $Da \rightarrow aaD,$

  $D] \rightarrow R],$

  $D] \rightarrow T,$

  $aR \rightarrow Ra,$

  $[R \rightarrow [D,$

  $aT \rightarrow Ta,$

  $[T \rightarrow \epsilon\}$

  generates the language $\{a^{2^n} : n \geq 0\}$.

- The variable $D$ is used to double the length of a string of $a$’s:

  

  $\quad Da^k \Rightarrow^* G_2 a^{2^k} D.$

- The symbols $[ \text{ and } ]$ are used as left and right markers between which the generation of a string $a^{2^k}$ takes place.

- The variable $R$ initiates another doubling via $D$, whereas $T$ terminates the process. The process is initialized via the rules for the start symbol $S$. 
Example

• The grammar $G_3 = (V_3, \Sigma_3, R_3, S)$ generates
  \[ \{ww : w \in \{a, b\}^*\} \].

• We have

  \[
  V_3 = \{S, T, R, L_a, L_b, A, B, a, b, [, ]\} \\
  \Sigma_3 = \{a, b\}, \quad \text{and} \\
  R_3 = \{S \to T, \\
    T \to aTA, \\
    T \to bTB, \\
    T \to [R, \\
    RA \to AR, RB \to BR, \\
    AR] \to L_a, BR] \to L_b, \\
    AL_a \to L_aA, AL_b \to L_bA, \\
    BL_a \to L_aB, BL_b \to L_bB, \\
    [L_a \to a[R, [L_b \to b[R, \\
    [R] \to \epsilon\} \]

• The rules for variables $S$ and $T$ can be used to generate any string $w[RW^R]$, where $w \in \{a, b\}^*$ and $W$ is obtained from $w$ by replacing lower-case by upper-case letters.

• The rules for variables $R$, $L_a$, and $L_b$ allow one to convert the string $[RW^R]$ to $w$, the brackets, $[ \text{ and } ]$, delimiting the substring still to be converted.
• The grammar $G_4 = (V_4, \{a, b\}, R_4, S_4)$ generates the language $\{a^n b^n a^n b^n : n \geq 0\}$, where

\[
\begin{align*}
V_4 &= \{S, A, B, C, V, X, Y, a, b\} \text{ and} \\
R_4 &= \{S \rightarrow aSBAC, S \rightarrow V, V \rightarrow \epsilon, \\
&\quad AB \rightarrow BA, \quad CB \rightarrow BC, \quad CA \rightarrow AC, \quad VB \rightarrow bV, \\
&\quad VA \rightarrowXA, \quad XA \rightarrow aX, \quad XC \rightarrow YC, \quad YC \rightarrow bY, \quad Y \rightarrow \epsilon\}.
\end{align*}
\]

• The first two rules allow us to generate any string of the form $a^nV(BAC)^n$. The third rule is needed for deriving the empty string.

• The next three rules can be used to transform such a string to $a^nVB^nA^nC^n$.

• The variable $V$ is used to control the conversion of occurrences of $B$ to $b$, and is eventually changed to $X$, which controls conversion of $A$ to $a$ and is eventually changed to $Y$, which in turn controls conversion of $C$ to $b$. 
We modify the grammar $G_3$ to a grammar $G_5$ for 
\[ \{ uvw : u, v \in \{a, b\}^*, u \neq v \}. \]

The idea is to use a slightly modified version of $G_3$ to derive strings of the form $wcw[R]$, and additional rules to then derive strings $wu'cwv'$, where $u'$ and $v'$ differ in the first symbol.

We have $G_5 = (V_5, \Sigma_5, R_5, S)$, where
\[
\begin{align*}
V_5 &= V_3 \cup \{X_a, Y_a, X_b, Y_b\}, \\
\Sigma_5 &= \{a, b, c\}, \text{ and} \\
R_5 &= (R_3 - \{T \rightarrow [R], [R] \rightarrow \epsilon\}) \cup \\
& \quad \{T \rightarrow c[R], \\
& \quad [R] \rightarrow X_aY_b, [R] \rightarrow X_a, [R] \rightarrow Y_a, \\
& \quad [R] \rightarrow X_bY_a, [R] \rightarrow X_b, [R] \rightarrow Y_b, \\
& \quad aX_a \rightarrow X_aa, bX_a \rightarrow X_ab, cX_a \rightarrow Ya, \\
& \quad aX_b \rightarrow X_ba, bX_b \rightarrow X_bb, cX_b \rightarrow Yb, \\
& \quad Y_a \rightarrow Ya, Y_a \rightarrow Yab, Y_a \rightarrow a, \\
& \quad Y_b \rightarrow Yba, Y_b \rightarrow Ybb, Y_b \rightarrow b\}.
\end{align*}
\]

Note that from $Y_a$ we can obtain any string in $\{a, b\}^*$ that begins with an $a$; from $Y_b$ any string that begins with a $b$. The variables $X_a$ and $X_b$ are used to insert $Y_a$ and $Y_b$, respectively, immediately to the left of $c$. 
Example

• The grammar $G_6$ generates the language

$$\{w^n : w \in \{a, b\}^*, |w| = n\}.$$  

• The idea is to generate first an arbitrary string $w$ of length $n$ and then to add $n - 1$ copies of $w$.

• We have $G_6 = (V_6, \Sigma_6, R_6, S)$, where

$$V_6 = \{S, T, D, X, Y, Z, R, L_a, L_b, [, ], a, b\}, \quad \Sigma_6 = \{a, b\}, \quad \text{and} \quad R_6 = \{S \to e, S \to a, S \to b, S \to T, T \to DT_a, T \to DT_b, T \to [a, T \to [b, D[\to X[R, Ra \to aR, Rb \to bR, R[\to [R, aR] \to L_a]a, bR] \to L_b]b, aL_a \to L_a a, bL_a \to L_a b, [L_a \to L_a[, aL_b \to L_b a, bL_b \to L_b b, [L_b \to L_b[, XL_a \to XaR, XL_b \to XbR, [R] \to Y[, [R] \to Z, aY \to Ya, bY \to Yb, XY \to [, aZ \to Za, bZ \to Zb, XZ \to \epsilon\}.}$$