1 General Grammars

Definition 1.1 (General Grammar) *A* (general) grammar *is a quadruple* (V, Σ, R, S) *where*

V	*is an alphabet*
$\Sigma \subseteq V$	*is the set of* terminal *symbols*
$V - \Sigma$	*is the set of* nonterminals
$S \in V - \Sigma$	*is the* start *symbol*
R	*is the set of* rules, *a subset of*

$$V^*(V - \Sigma)V^* \quad \times \quad V^*$$

left-hand side at least one nonterminal	*right-hand side may have anything*

Note that context-free grammars are also (general) grammars. As for derivations, $w_1uw_2 \Rightarrow_G w_1vw_2$ for $w_1, w_2 \in V^*$ if $(u, v) \in R$. Then $\Rightarrow_G^*$ is defined as a sequence of zero or more $\Rightarrow_G$. Also, $w \in \Sigma^*$ is *generated by* G iff $S \Rightarrow^* w$ and $L(G) = \{w \in \Sigma^* : w \text{ is generated by } G\}$. A *derivation* in G is a sequence

$$w_o \Rightarrow_G w_1 \Rightarrow_G w_2 \Rightarrow_G \ldots \Rightarrow_G w_n.$$

General grammars can be seen as having *context-dependence* which means that a replacement can only be applied in a specified context. A production of the form

$$uAv \rightarrow uwv$$

can be seen as applying the replacement $A \rightarrow w$ only when u is to the left and v is to the right. Any general grammar can be simulated using only productions of this form, it turns out. This helps to motivate the term "context-free."

General grammars are more powerful than context-free grammars. Here is a grammar whose language is $\{a^n b^n c^n : n \geq 0\}$, which we know is not context-free:

$$
\begin{aligned}
G &= (V, \Sigma, R, S) \\
V &= \{S, a, b, c, A, B, C, T_a, T_b, T_c\} \\
\Sigma &= \{a, b, c\}
\end{aligned}
$$

$$
R = \begin{array}{ll}
\{S \rightarrow ABCS & \text{Generate } (ABC)^n T_c \\
S \rightarrow T_c & \\
 & \\
CA \rightarrow AC & \text{Sort} \\
BA \rightarrow AB & \\
CB \rightarrow BC & \\
 & \\
CT_c \rightarrow T_c c & \text{change } C \text{ to } c \\
CT_c \rightarrow T_b c & \\
 & \\
BT_b \rightarrow T_b b & \text{change } B \text{ to } b \\
BT_b \rightarrow T_a b & \\
 & \\
AT_a \rightarrow T_a a & \text{change } A \text{ to } a \\
T_a \rightarrow \epsilon\} &
\end{array}
$$

Here is a derivation of $aabbcc$ in this grammar:

$$S \Rightarrow ABCS \Rightarrow ABCABCS \Rightarrow ABCABCT_c$$

$$\Rightarrow ABACBCT_c \Rightarrow AABCBCTc \Rightarrow AABBCCT_c$$

$$\Rightarrow AABBCT_c c \Rightarrow AABBT_b cc \Rightarrow AABT_b bcc$$

$$\Rightarrow AAT_a bbcc \Rightarrow AT_a abbcc \Rightarrow T_a aabbcc \Rightarrow aabbcc$$

Theorem 1.1 *A language is generated by a grammar iff it is recursively enumerable.*

This shows that grammars are equivalent in power to Turing machines, in some sense. The proof simulates a grammar by a Turing machine and a Turing machine by a grammar. It is straightforward, if cumbersome, to design a Turing machine to simulate a grammar. It also fairly easy to simulate a Turing machine by a grammar.